

Cours 2

Performance d'un classifieur

Méthode supervisée vs non supervisée

Méthode générative vs discriminative Classification
par les kppv

Catherine ACHARD

Institut des Systèmes Intelligents et de Robotique

catherine.achard@sorbonne-universite.fr

Classifier

= associer une **classe** C

à un **vecteur de caractéristiques** x de dimension n

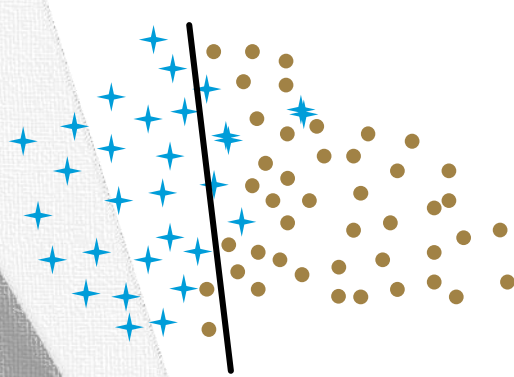
Base de données divisée en 3

- **Base d'apprentissage** : pour apprendre le modèle
- **Base de validation** : pour aider à définir certains paramètres du modèle
- **Base de test** : pour tester le modèle sur de nouvelles données jamais rencontrées

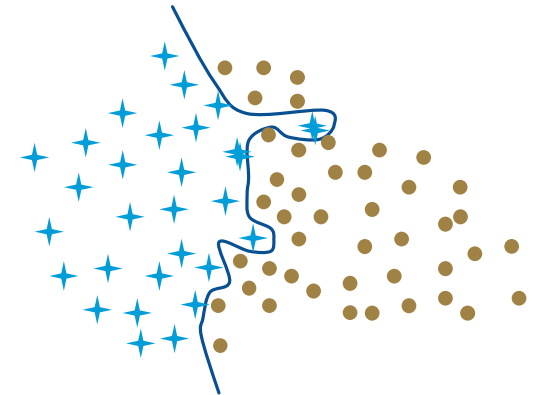
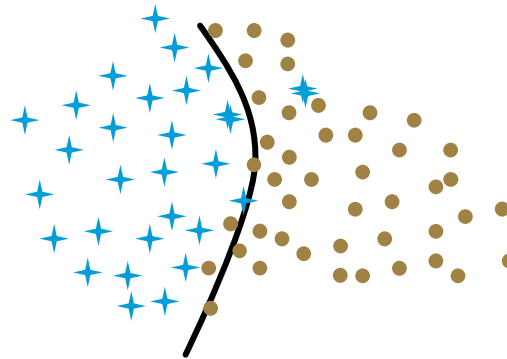
Un bon apprentissage devra

- Avoir une faible erreur sur la base d'apprentissage
- Avoir un écart faible entre l'erreur d'apprentissage et l'erreur de test

Lorsque c'est le cas, on parle de **bonne généralisation** : le système est capable de reconnaître des données jamais vues



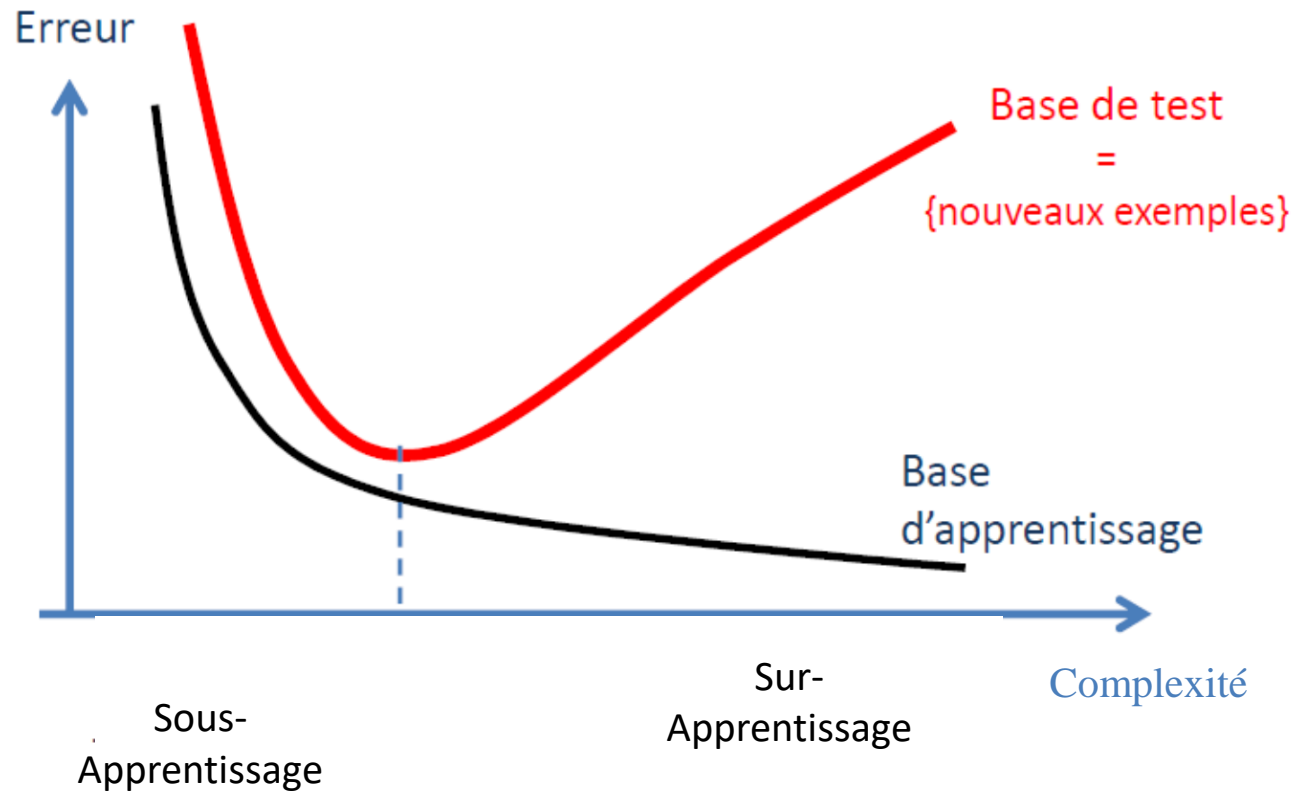
Sous-apprentissage



Sur-apprentissage

Généralisation

- Bonne généralisation, où est la frontière ?



Sur-apprentissage (over-fitting)

On se fie trop aux données

- petit changement de la base d'apprentissage = forte variation de la sortie
- Variance élevée, biais faible

Sous-apprentissage (under-fitting)

On se fie moins aux données

- on s'éloigne par endroit de la vraie frontière
- **Biais élevé, variance faible**

Compromis biais/variance

Comment avoir un bon compromis biais/variance ?

Comment avoir une bonne généralisation

- En diminuant la dimension des exemples
- En augmentant le nombre d'exemples
- Sélectionner un bon modèle ou en le contraignant
- Méthodes ensemblistes (il vaut souvent mieux recueillir plusieurs avis que de se fier à un seul → on apprend sur plusieurs ensembles de données puis vote majoritaire par exemple)

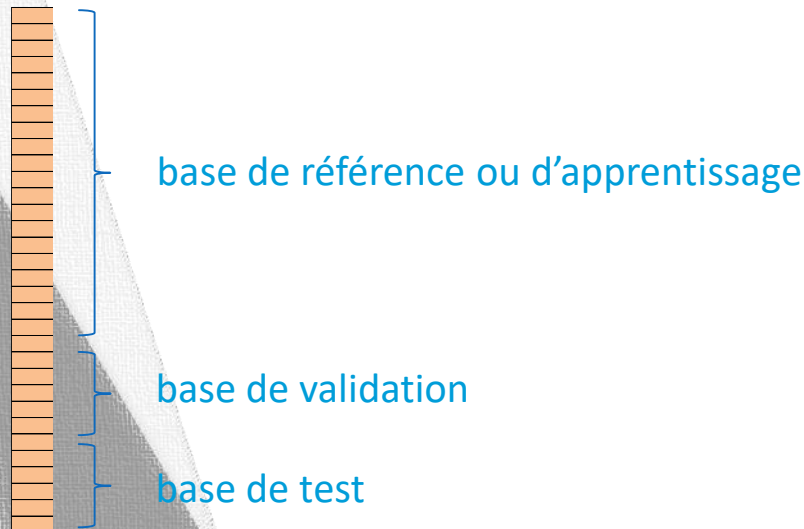
Performances d'un classifieur

En apprentissage, il n'y a **pas de méthode idéale**, celle-ci varie en fonction des données, de leur dimension, du problème posé,...

- Le choix de la méthode la plus adaptée à un cas précis repose sur une **estimation de l'erreur**
- Cette estimation devra être la plus rigoureuse possible

On utilise 3 bases :

- **Une base de référence ou d'apprentissage** utilisée pour mettre au point le classificateur
- **Une base de validation** pour déterminer les paramètres du classifieur
- **Une base de test** : exemples jamais vus au préalable pour évaluer le classificateur



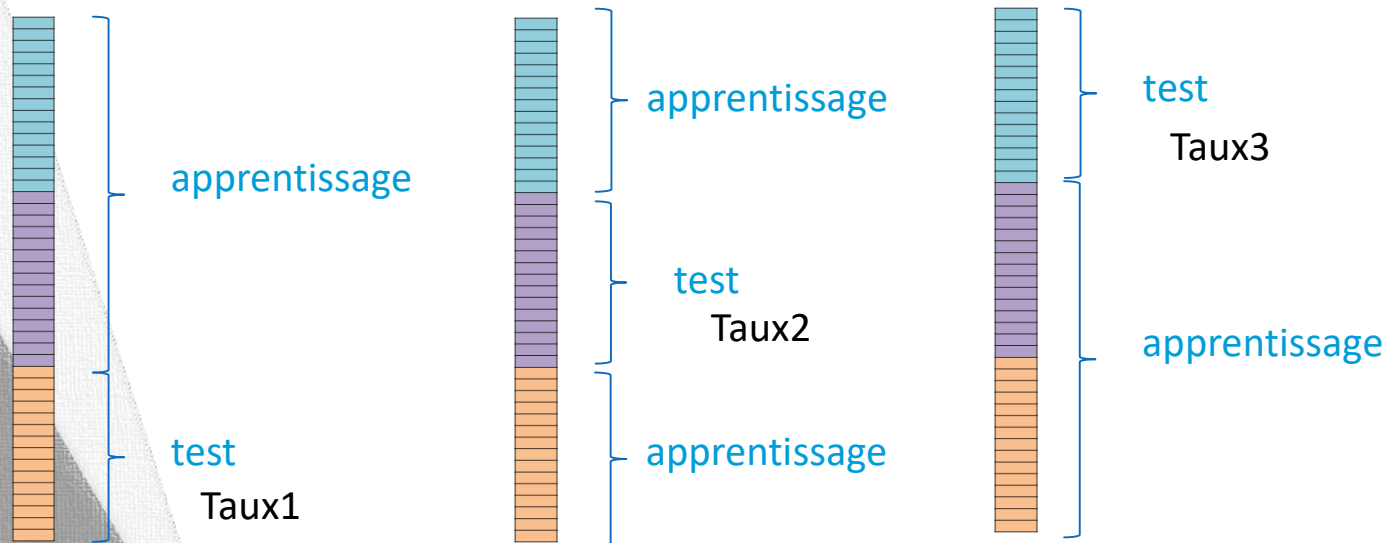
Petite base de données

Avec peu de données, il est difficile d'estimer les performances de manière fiable.

Solution 1 : K-fold validation :

- Division de la base en K parties.
- (K-1) parties pour l'apprentissage et la validation, la dernière en test
- On fait tourner la partie test sur les K parties
- Le taux de reconnaissance final est la moyenne des taux

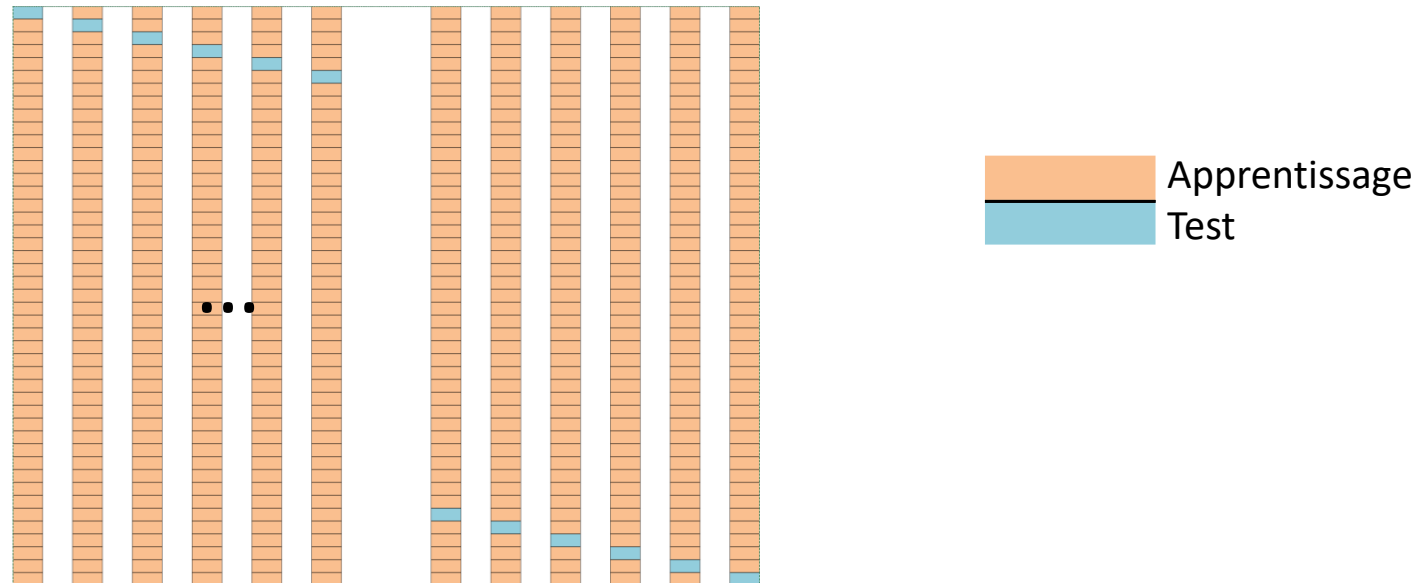
Exemple avec K=3



$$\text{Taux} = (\text{Taux1} + \text{Taux2} + \text{Taux3})/3$$

Solution 2 : Leave-One-Out Cross-Validation (LOOCV)

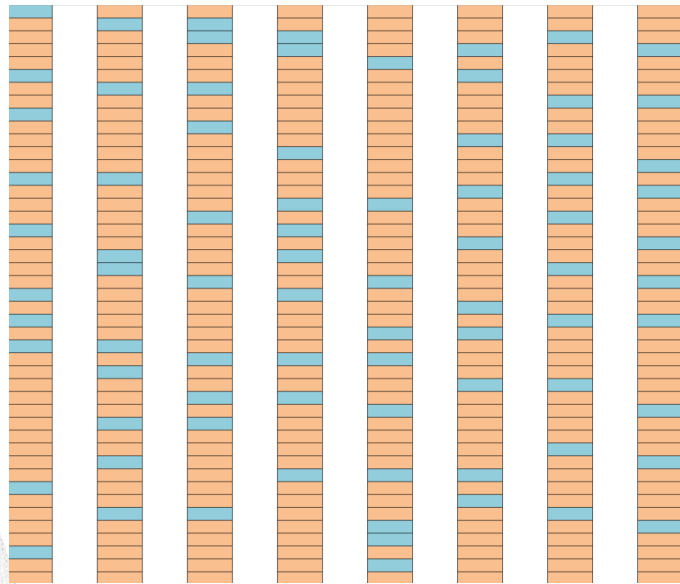
- C'est une K-fold validation dans le cas particulier où $K = \text{nombre d'exemples}$
- $N-1$ exemples pour l'apprentissage et la validation, le dernier exemple en test
- On fait tourner l'exemple de test sur tous les exemples de la base



- Il faut faire autant d'apprentissages qu'il y a d'exemples de test → très long
- Taux de reco = $\text{Nb d'exemples bien classés} / \text{Nb d'exemples}$

Solution 3 : bootstrap

- Tirage aléatoire pour déterminer les exemples d'apprentissage et de test
- Plusieurs itérations puis taux moyen



Apprentissage
Test

- A chaque fois on calcule le taux de reconnaissance sur les exemples de test
- Le taux de reconnaissance final est le taux moyen

Matrice de confusion :

Classe réelle
↓

Classe trouvée →

	1	2	3
1	e_{11} Nb d'exemples 1 étiquetés 1	e_{12} Nb d'exemples 1 étiquetés 2	e_{13} Nb d'exemples 1 étiquetés 3
2	e_{21} Nb d'exemples 2 étiquetés 1	e_{22} Nb d'exemples 2 étiquetés 2	e_{23} Nb d'exemples 2 étiquetés 3
3	e_{31} Nb d'exemples 3 étiquetés 1	e_{32} Nb d'exemples 3 étiquetés 2	e_{33} Nb d'exemples 3 étiquetés 3

Exercice

On donne la matrice de confusion ci-dessous :

Classe trouvée →

Classe réelle ↓

	1	2	3
1	90	6	5
2	20	70	4
3	2	1	104

Quel est le nombre d'exemples de la base de test ?

Que représente le chiffre 20 ?

Que représente le chiffre 104 ?

Quel est le taux de bonne reconnaissance ?

Exercice

On donne la matrice de confusion ci-dessous :

Classe réelle
↓

Classe trouvée →

	1	2	3
1	90	6	5
2	20	70	4
3	2	1	104

Quel est le nombre d'exemples de la base de test ?

La somme des éléments de la matrice = 302

Que représente le chiffre 20 ?

Le nombre d'exemples appartenant à la classe 2 étiquetés 1

Que représente le chiffre 104 ?

Le nombre d'exemples appartenant à la classe 3 étiquetés 3

Quel est le taux de bonne reconnaissance ?

La somme des éléments de la diagonale divisée par la somme des éléments
 $264/302 \rightarrow 87,4 \%$

Taux de bonne classification

Sans rejet

$$\text{Taux de bonne classification } T_b = \frac{\text{Nb exemples bien classés}}{\text{Nb exemples}}$$

$$\text{Taux d'erreur } T_e = 1 - T_b$$

Avec rejet

$$\text{Taux de rejet } T_r = \frac{\text{Nb exemples rejeté}}{\text{Nb exemples}}$$

$$\text{Taux de bonne classification } T_b = \frac{\text{Nb exemples bien classés}}{\text{Nb exemples}}$$

$$\text{Taux d'erreur } T_e = 1 - T_b - T_r$$

Taux de bonne classification

Problème :

Le taux de bonne classification est une mesure faible qui **ne tient pas compte de la distribution des classes**

➔ Quand les classes sont très disproportionnées, prédire systématiquement la classe majoritaire amène à un bon taux de classification

Exercice :

En diagnostic médical, très peu de personnes sont malades (5%?). Quel est le taux de classification si on prédit toujours que la personne est saine ?

Exemple sur 100 personnes

	malade	sain
malade	0	5
sain	0	95

Tb=95%

Solution :

On tient compte de la répartition des classes et on construit une **matrice de confusion normalisée**

	e_{11}/N_1	e_{12}/N_1
	e_{21}/N_2	e_{22}/N_2

N_1 : Nombre d'exemples de la classe 1

N_2 : Nombre d'exemples de la classe 2

Le nouveau taux de bonne classification devient

$$tb = \frac{1}{N_c} \sum_{i=1}^{N_c} \frac{e_{ii}}{N_i} \text{ où } N_c \text{ est le nombre de classes}$$

Exercice :

reprendre l'exemple précédant avec cette normalisation

On a maintenant

	malade	sain
malade	0	1
sain	0	1

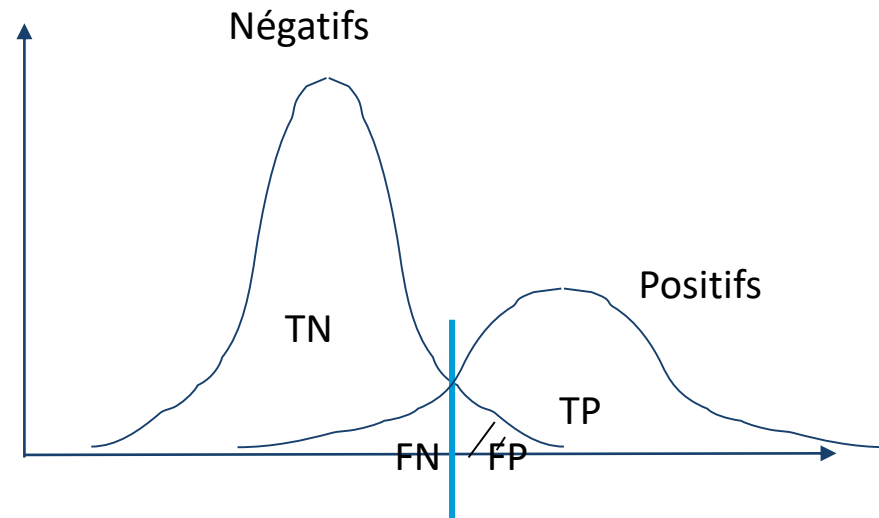
Et $t_b = 50\%$

Courbe ROC (Receiver Operating Characteristic)

Permet de comparer plusieurs classificateurs indépendamment d'un seuil pour un problèmes à 2 classes (**positif et négatif**)

On définit les :

- Vrai Positif (True Positive)
- Vrai Négatif (True Négatif)
- Faux Négatif (False Négatif)
- Faux Positif (False Positif)



Classe réelle



Classe trouvée



	+	-
+	TP	FN
-	FP	TN

Courbe ROC (Receiver Operating Characteristic)

$$\text{Sensibilité} = \frac{TP}{TP+FN} = \frac{\text{Nb positifs bien classés}}{\text{Nb positifs}}$$

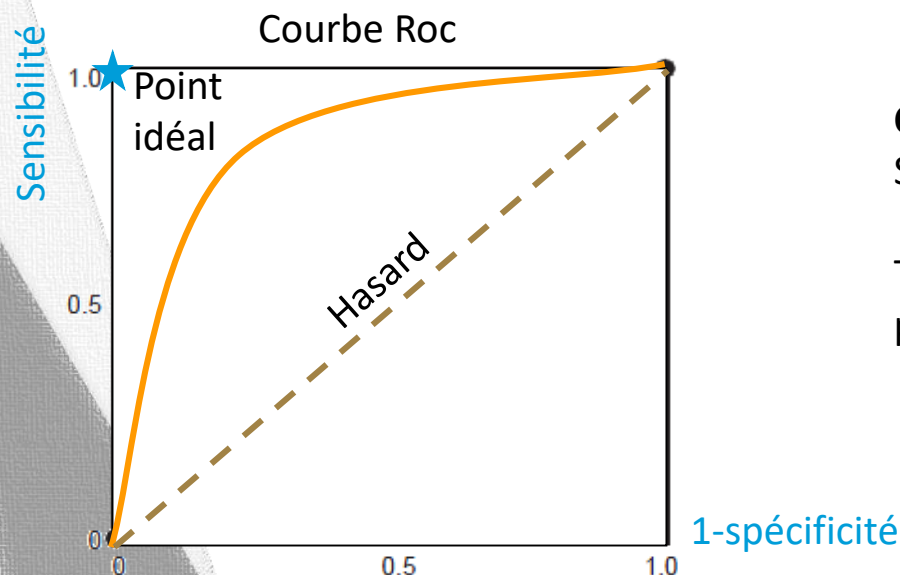
$$\text{Spécificité} = \frac{TN}{FP+TN} = \frac{\text{Nb négatifs bien classés}}{\text{Nb négatifs}}$$

	+	-
+	TP	FN
-	FP	TN

Un bon classificateur devra être

- sensible : détecter les positifs = pourcentage de vrais positifs détectés
- spécifique : ne pas tout détecter comme positif = pourcentage de vrais négatifs détectés

Généralement, plus un classificateur est sensible, moins il est spécifique et vice versa



Courbe ROC

$$\text{Sensibilité} = f(1-\text{spécificité})$$

Toutes les courbes ROC passent par l'origine et par le point (1,1)

Précision/Rappel

On trouve aussi parfois les scores exprimés en termes de Précision / Rappel

Précision = $\frac{TP}{TP+FP} = \frac{\text{Nb positifs bien classés}}{\text{Nb classés positifs}}$ = pourcentage de vrais positifs parmi les positifs trouvés

Rappel $\frac{TP}{TP+FN} = \frac{\text{Nb positifs bien classés}}{\text{Nb positifs}}$ = pourcentage de vrais positifs détectés (même chose que sensibilité)

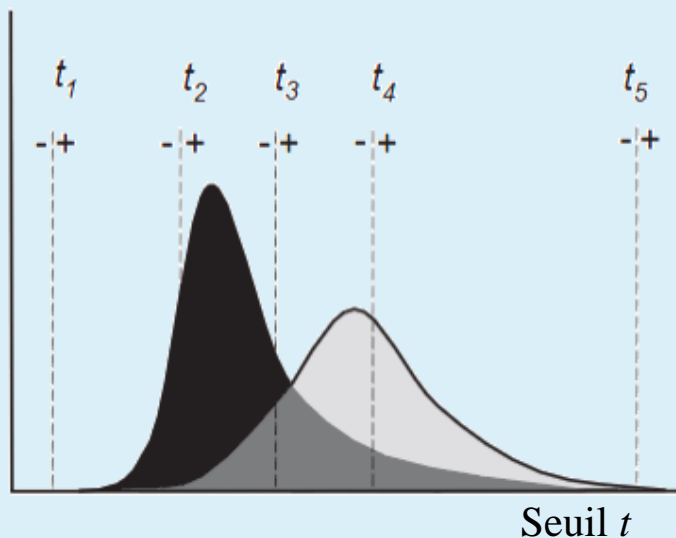
	+	-
+	TP	FN
-	FP	TN

Exercice

On souhaite réaliser une classification binaire. Plusieurs algorithmes dépendant d'un seuil doivent être comparés.

Un algorithme donné donne les densités de probabilité de chaque classe en fonction du seuil.

Tracer la courbe ROC correspondante

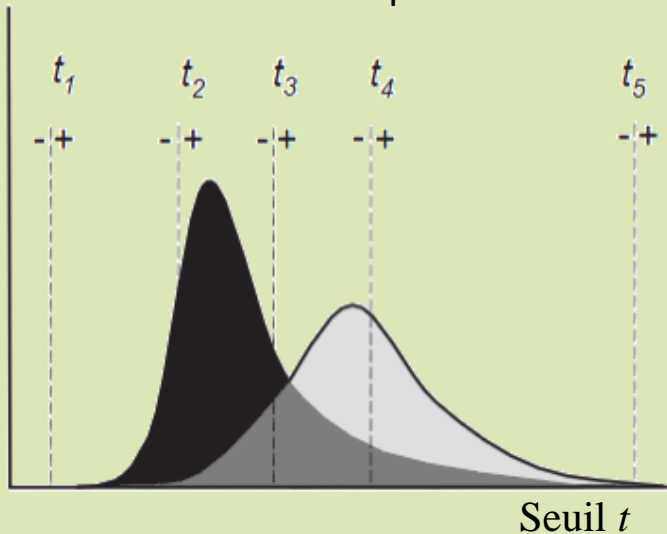


Exercice

On souhaite réaliser une classification binaire. Plusieurs algorithmes dépendant d'un seuil doivent être comparés.

Un algorithme donné donne les densités de probabilité de chaque classe en fonction du seuil.

Tracer la courbe ROC correspondante

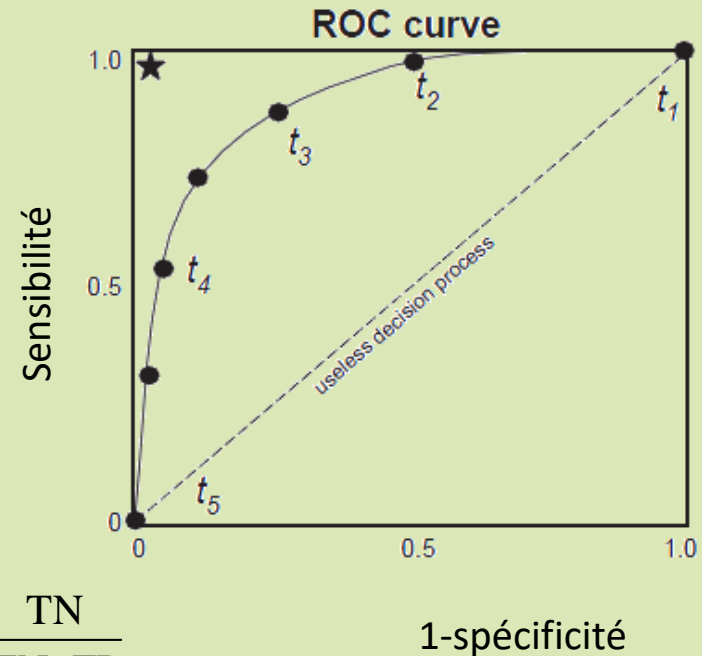


$$\text{Sensibilité} = \frac{TP}{TP + FN}$$

$$\text{Spécificité} = \frac{TN}{TN + FP}$$

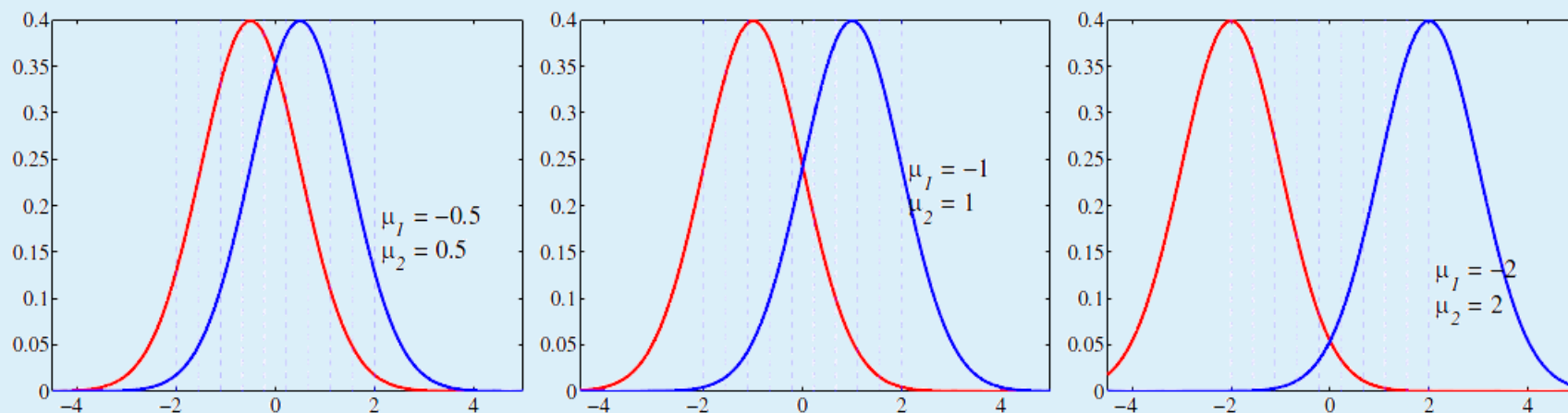
Remarque:

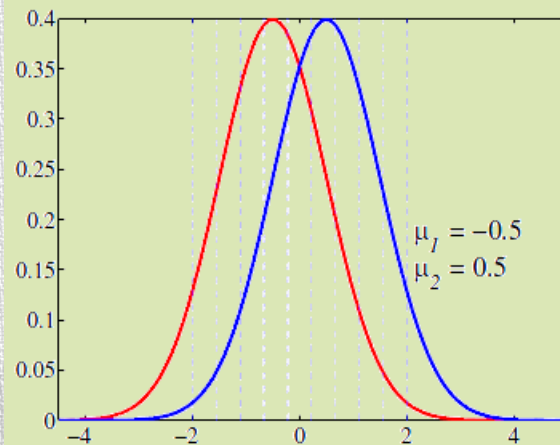
$$\begin{cases} TP + FN = cte \\ TN + TP = cte \end{cases}$$



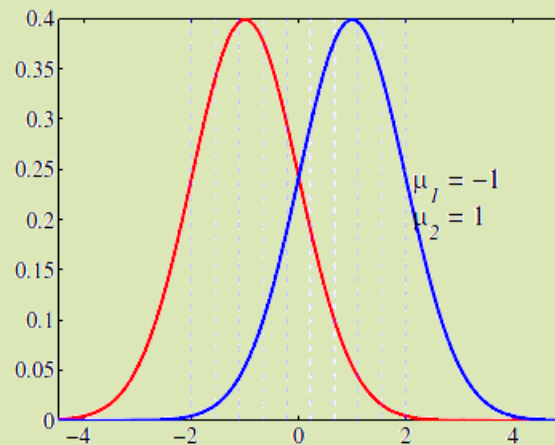
Exercice

Tracer l'allure des courbes ROC des 3 classifieurs ci-dessous (mêmes figures que pour l'exercice précédent))

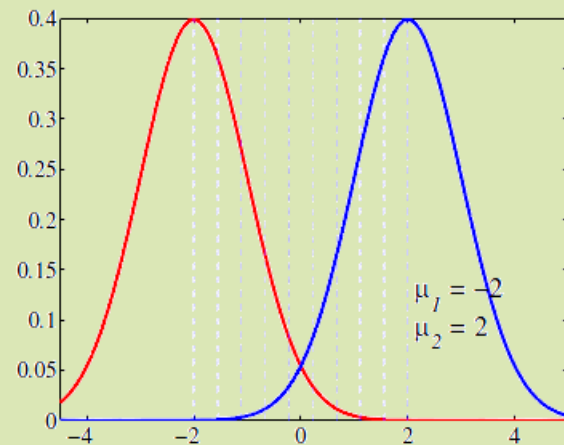




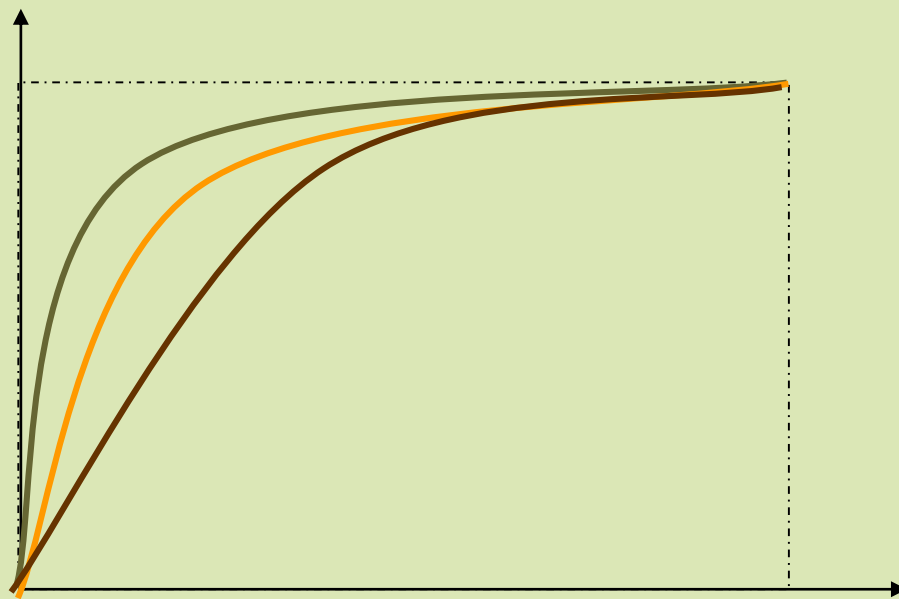
Classifieur 1



Classifieur 2



Classifieur 3



Exercice

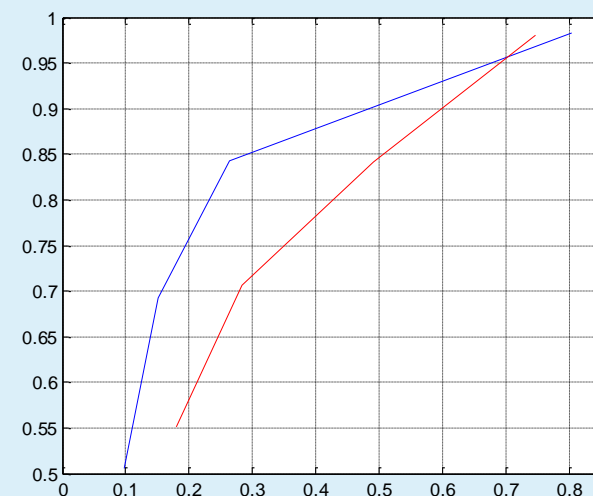
Nous souhaitons détecter des objets en mouvement (positif) dans des séquences d'images.

→ 2 algorithmes que l'on veut comparer indépendamment du seuil de détection

A = Nombre de pixels détectés en mouvement
B = Nombre de pixels détectés non mouvement
C = Nombre de pixels réellement en mouvement et détecté en mouvement
D = Nombre de pixels réellement en mouvement = 57587

		A	B	C
Première méthode	Seuil1	372417	77583	56629
	Seuil2	151899	298101	48526
	Seuil3	98997	351003	39847
	Seuil4	67181	382819	29135
Deuxième méthode	Seuil1	349806	100194	56479
	Seuil2	241185	208815	48472
	Seuil3	151902	298098	40654
	Seuil4	102264	347736	31739

1. Donner l'expression analytique du nombre de vrais positifs (TP), vrais négatifs (TN), faux positifs (FP) et faux négatifs (FN) en fonction de A, B, C, D.
2. Dans la base de test, quel est le nombre d'exemples positifs et d'exemples négatifs ? Donner l'expression analytique et le résultat numérique.
3. Donner l'expression analytique de la sensibilité et de la spécificité en fonction de A, B, C et D.
4. Les deux courbes ROC sont représentées ci-contre. Quelle est la courbe ROC de la première méthode ? Justifier votre réponse. Quelle méthode donne les meilleurs résultats ?



1. Donner l'expression analytique du nombre de vrais positifs (TP), vrais négatifs (TN), faux positifs (FP) et faux négatifs (FN) en fonction de A, B, C, D.

décision \ étiquette	+	-
+	TP	FN
-	FP	TN

$$\begin{aligned}
 A &= TP + FP \\
 B &= TN + FN \\
 C &= TP \\
 D &= TP + FN
 \end{aligned}
 \Rightarrow
 \begin{cases}
 TP = C \\
 FP = A - C \\
 FN = D - C \\
 TN = B - D + C
 \end{cases}$$

2. Dans la base de test, quel est le nombre d'exemples positifs et d'exemples négatifs ?
Donner l'expression analytique et le résultat numérique.

$$\text{Nb ex} > 0 = TP + FN = C + D - C = D = 57587$$

$$\text{Nb Ex} < 0 = TN + FP = A - C + B - D + C = A + B - D = 392413$$

3. Donner l'expression analytique de la sensibilité et de la spécificité en fonction de A, B, C et D.

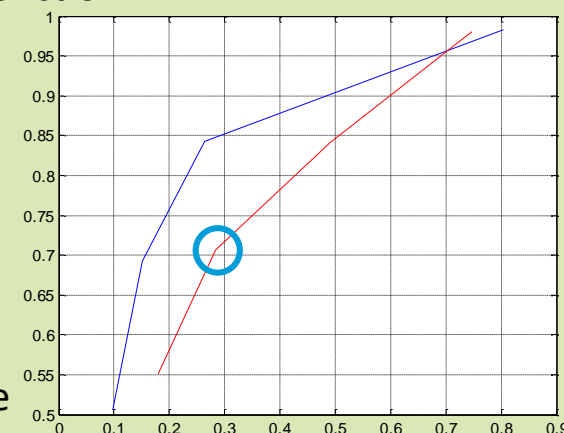
$$\text{sens} = \frac{C}{D} = \frac{C}{57587} \quad \text{spe} = \frac{B - D + C}{A + B - D} = \frac{B - D + C}{392413}$$

En prenant le seuil 3 de la méthode 2, on a :

Sens=0.7 et Spe=0.71 \rightarrow 1-spe=0.28

\rightarrow c'est la courbe du bas

\rightarrow La courbe du haut est la première méthode et celle du bas la seconde



Méthodes supervisées / Non supervisées

Méthode supervisée

On dispose d'un **ensemble d'exemples étiquetés** en apprentissage

➔ On souhaite savoir classer un nouvel exemple

Le nombre de classes est connu *a priori*

Exemple

On dispose d'un ensemble de signaux audio enregistré à partir de 50 personnes. L'identité de la personne est connue pour chaque enregistrement

Pour un nouveau signal, on souhaite déterminer l'identité de la personne

Méthode non supervisée

On dispose d'un **ensemble d'exemples non étiquetés** en apprentissage

→ On souhaite partitionner cet ensemble en sous-ensembles homogènes

Le nombre de classes (sous-ensembles) n'est pas connu *a priori*

Exemple

On dispose des achats de livres de clients sur amazon

On souhaite catégoriser les clients en fonction de leur goûts afin de leur proposer des nouveaux achats pertinents

→ On peut utiliser ces méthodes pour explorer et comprendre les données

Méthodes génératives / discriminatives

Soit \mathbf{x} les exemples de dimension n et y leur classe telle que $y \in [1, \dots, K]$

But de la classification : déterminer $p(y|\mathbf{x})$

Approche discriminative

Détermine directement $p(y|\mathbf{x})$

Approche générative / Classification bayésienne :

Détermine pour chaque classe $p(\mathbf{x}|y)$ et $P(y)$ puis utilise le théorème de Bayes :

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

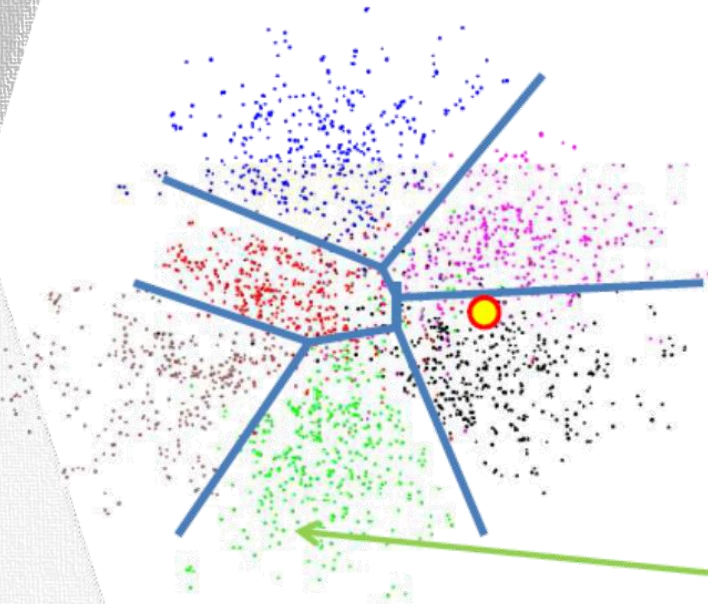
Où le dénominateur est un terme de normalisation :

$$p(\mathbf{x}) = \sum_y p(\mathbf{x}|y)p(y)$$

Cette approche est dite générative car, connaissant $p(\mathbf{x}|y)$, il est facile de générer des données dans l'espace des paramètres

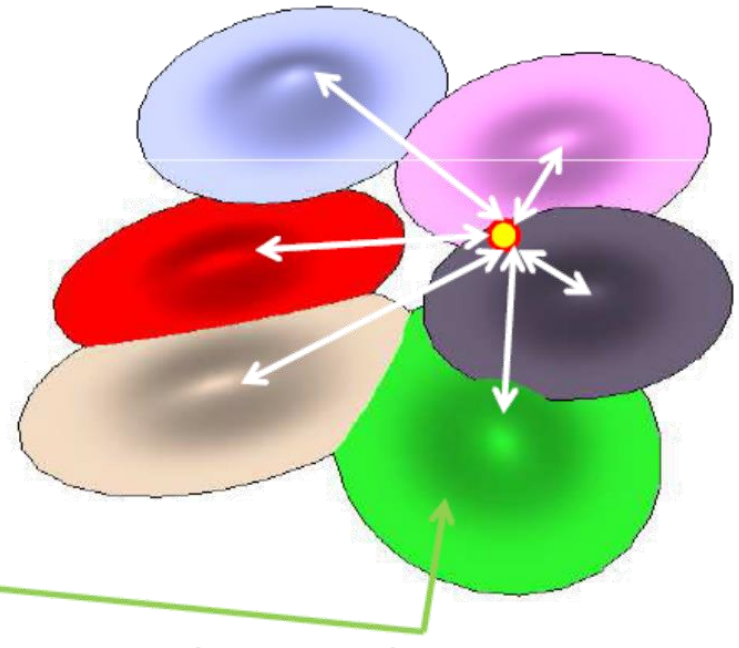
Approche discriminante

On apprend les frontières entre les classes



Approche générative

On modélise chaque classe




Exemple : on souhaite déterminer la langue parlée par une personne


Approche générative : on apprend chaque langue puis on détermine à quel langue la parole appartient (peut fonctionner avec une seule langue pour savoir si la personne parle français ou non)

Approche discriminative: on apprend les différences linguistiques entre les langues, sans apprendre la langue. Beaucoup plus simple !



Avantage/inconvénient

Approche générative

- 
- $p(x|y)$ est estimée. On peut considérer $p(x|y)$ comme la probabilité que x soit bien modélisé par le modèle. Ceci permet de faire du rejet ou de combiner des classifieurs.
 - $p(x|y)$ peut être utilisé pour générer des données
 - Permet à un système d'utiliser une seule classe. Ex : la teinte chair

- 
- Trouver $p(x|y)$ pour chaque classe est très coûteux en temps de calcul, surtout quand x est de grande dimension
 - Nécessite une grande base de données, surtout quand x est de grande dimension

Approche discriminative

- 
- Il est beaucoup plus rapide de déterminer $p(y|x)$ car la dimension de y est bien plus faible que celle de x
- 
- On ne peut pas générer de données
 - On ne peut pas faire de rejet
 - Difficile de combiner des classifieurs

Méthodes génératives	Méthodes discriminatives
Classification bayésienne	K plus proches voisins
HMM (Hidden Markov Model)	Arbres de décision
Réseaux bayésiens	SVM (Support Vector Machine)
MRF (Markov Random Fields)	RVM (Relevance Vector Machine)
	Réseaux de neurones
	CRF (Conditional Random fields)

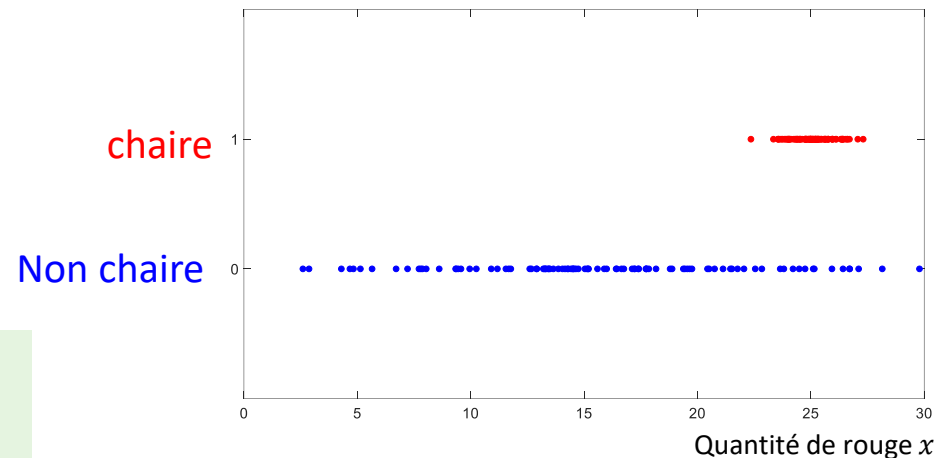
Exemple : classification binaire

exemple issu de computer vision: models, learning and inference, Simon J.D. Prince 2012

On souhaite classer des pixels en teinte chaire ou non chaire à partir de la quantité de rouge. On dispose de 200 pixels de teinte chaire et 800 de teinte non chaire

x est une variable continue (quantité de rouge)

2 classes : teinte chaire $y=1$ ou non chaire $y=0$



Loi de Bernoulli

Pour x une variable binaire,

$$\text{Bern}(x = 1) = \mu \text{ et}$$

$$\text{Bern}(x = 0) = 1 - \mu$$

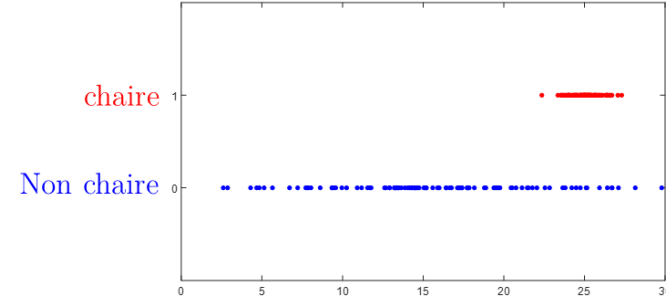
et de manière générale

$$\text{Bern}(x) = \mu^x (1 - \mu)^{1-x}$$

Approche générative

Modélisation :

- On modélise $p(x|y = 0)$ et $p(x|y = 1)$ par des gaussiennes
 - On utilise les données d'apprentissage de la classe 0 pour estimer (μ_0, σ_0) et celles de la classe 1 pour estimer (μ_1, σ_1)
 - $\mu_0 = 15$, $\sigma_0 = 6$, $\mu_1 = 25$, $\sigma_1 = 1$
- On modélise $p(y)$ par une loi de Bernoulli de paramètre μ
 - $\mu = 0.2$ (pourcentage de pixels de la classe 1).
 - On a donc $p(y = 0) = 0.8$ et $p(y = 1) = 0.2$



Tous les paramètres $(\mu_0, \sigma_0, \mu_1, \sigma_1, \mu)$ ont été appris sur la base d'apprentissage

Classification: Un nouveau pixel x arrive et on souhaite le classer. On estime :

- On $p(x|y = 0) = \frac{1}{\sigma_0\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_0)^2}{2\sigma_0^2}\right)$ et $p(x|y = 1) = \frac{1}{\sigma_1\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_1)^2}{2\sigma_1^2}\right)$

- Puis on estime $p(y|x)$ en utilisant Bayes :

$$p(y = 0|x) = \frac{p(x|y=0)p(y=0)}{p(x|y=0)p(y=0) + p(x|y=1)p(y=1)}$$

$$p(y = 1|x) = \frac{p(x|y=1)p(y=1)}{p(x|y=0)p(y=0) + p(x|y=1)p(y=1)}$$

Et on garde la plus grande valeur

Approche discriminative

Modélisation :

- On modélise $p(y|x)$ par une loi de Bernouilli $p(y = 1|x) = \mu$ dont le paramètre μ dépend de x .

Comme $0 < \mu < 1$, on pose $\mu = \frac{1}{1+\exp(-\Phi_0-\Phi_1x)}$. On utilise les données d'apprentissage (x_i, y_i) pour estimer les paramètres (Φ_0, Φ_1)

$$\Phi_0 + \Phi_1 x_i = \ln\left(\frac{\mu}{1-\mu}\right) \Rightarrow \text{relation linéaire}$$

On estime les paramètres de la droite la droite $\Phi_0 + \Phi_1 x_i = y_i$ en utilisant la méthode des moindres carrés par exemple

(Φ_0, Φ_1) appris sur la base d'apprentissage $\Phi_0 = -42.6$, $\Phi_1 = 1.68$

Classification: Un nouveau pixel x arrive et on souhaite le classer. On estime directement :

$$p(y = 1|x) = \frac{1}{1 + \exp(-\Phi_0 - \Phi_1 x)}$$

Algorithme des kppv

KPPV : K plus proches voisins

Il s'agit d'une **méthode supervisée et discriminative**

Données de départ

Ensemble d'exemples de dimensions n étiquetés $y \in [1, \dots, K]$

But

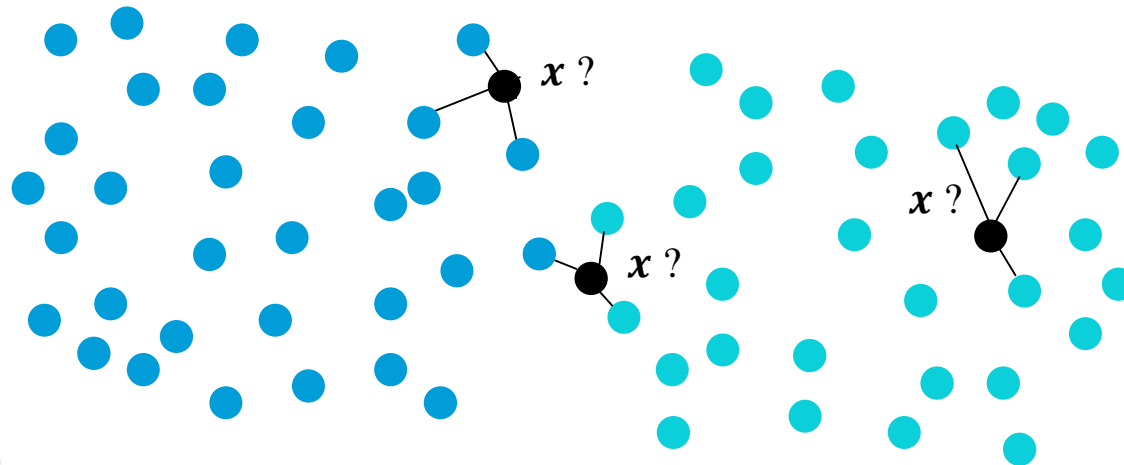
Trouver la classe d'un nouvel exemple x

Méthode

- Calculer la distance entre x et tous les exemples de la base de référence
- Déterminer les K exemples les plus proches.
- Affecter à x la classe majoritaire parmi les K plus proches voisins

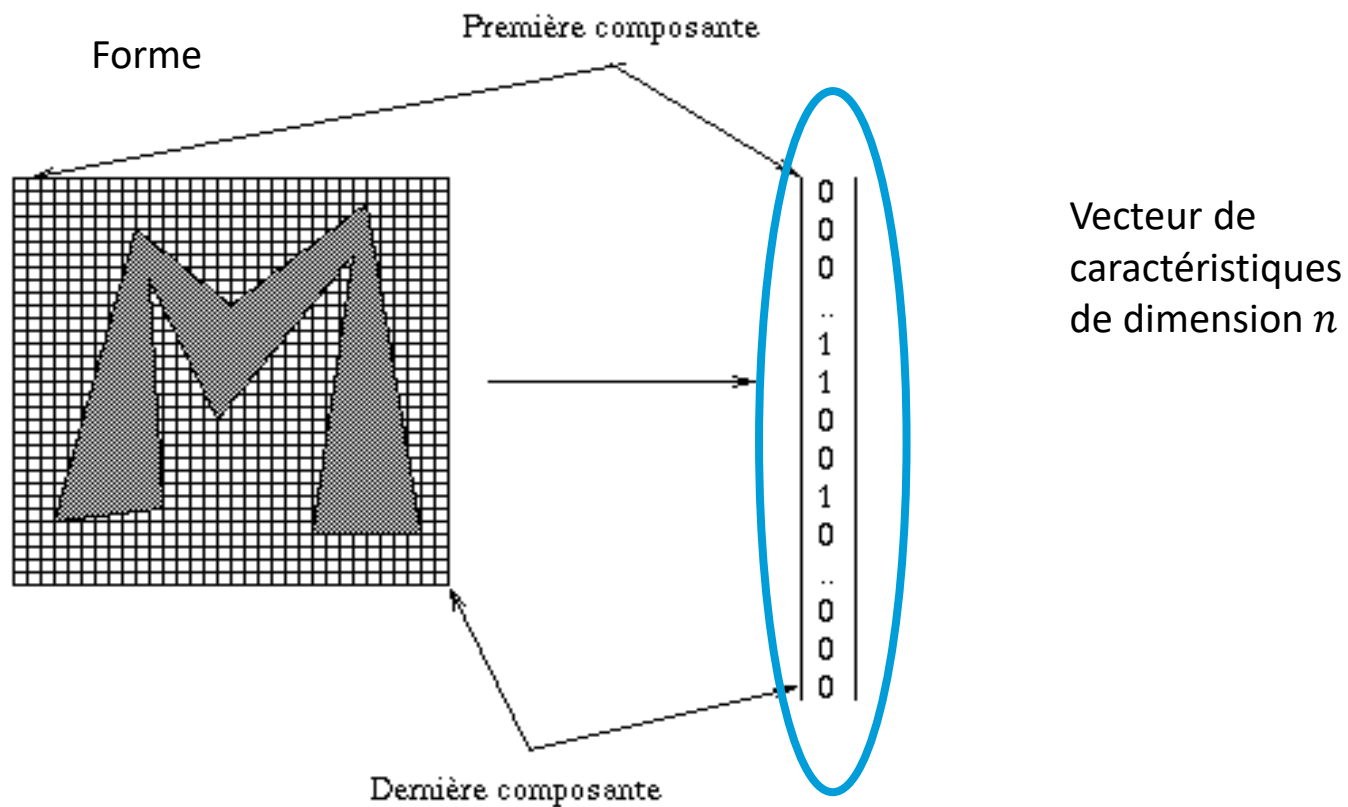
Exemple

En dimension 2, chaque exemple est caractérisé par un vecteur de dimension 2 et est donc représenté dans le plan :

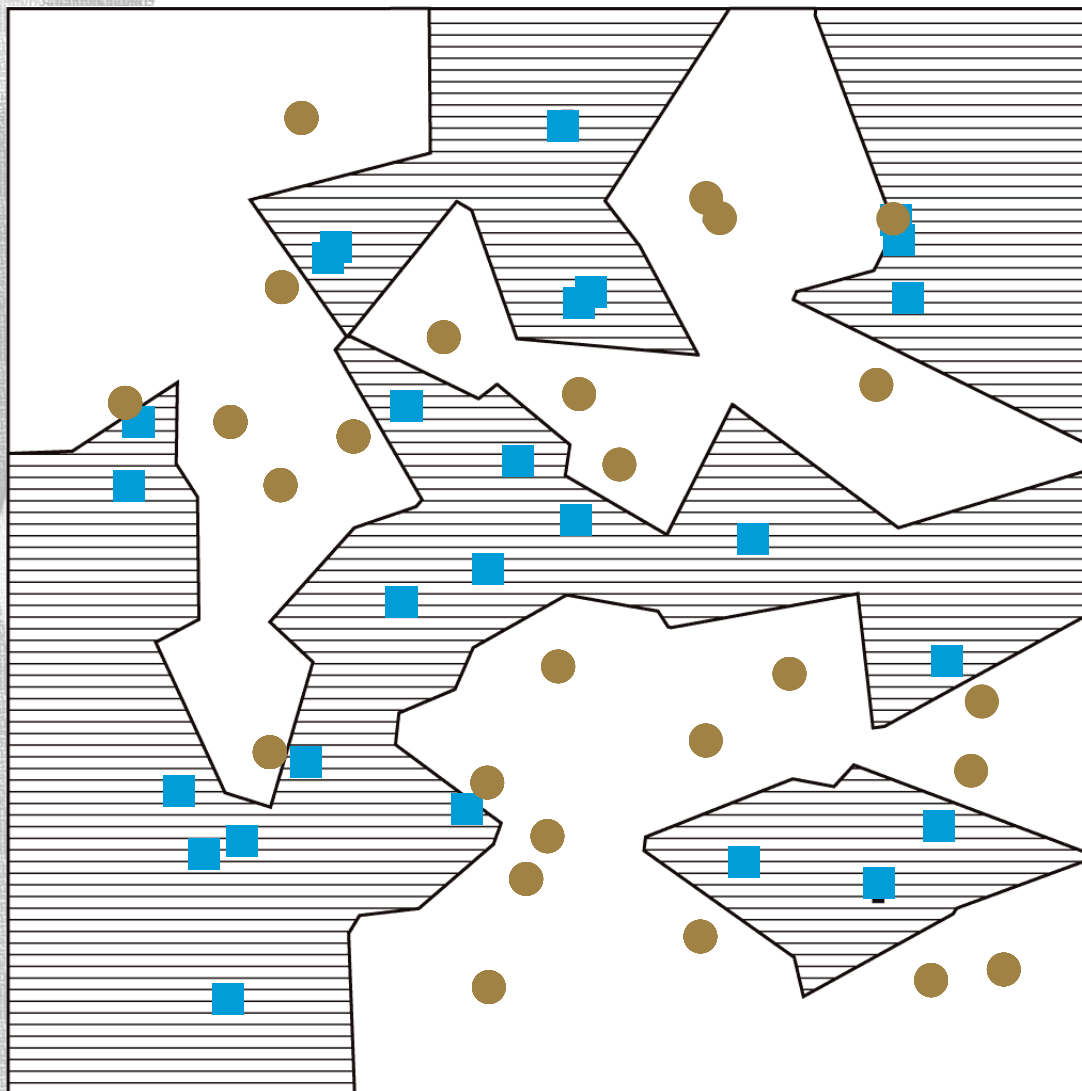


Exemple

En dimension n



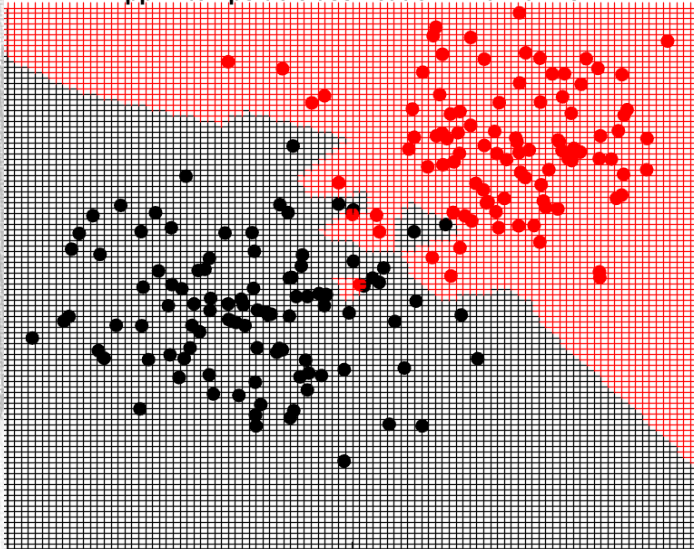
Signification géométrique 1ppv



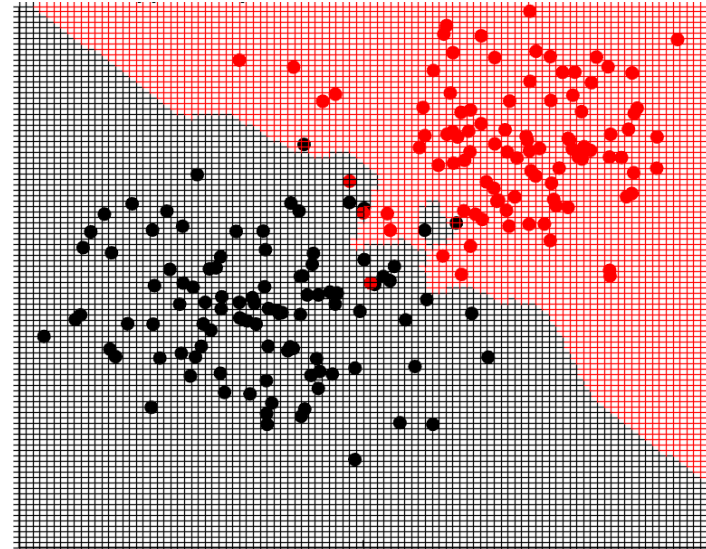
Les classes sont définies par la réunion des domaines d'influence des exemples de références

La résolution spatiale des frontières est liée au nombre d'exemples et à leur densité

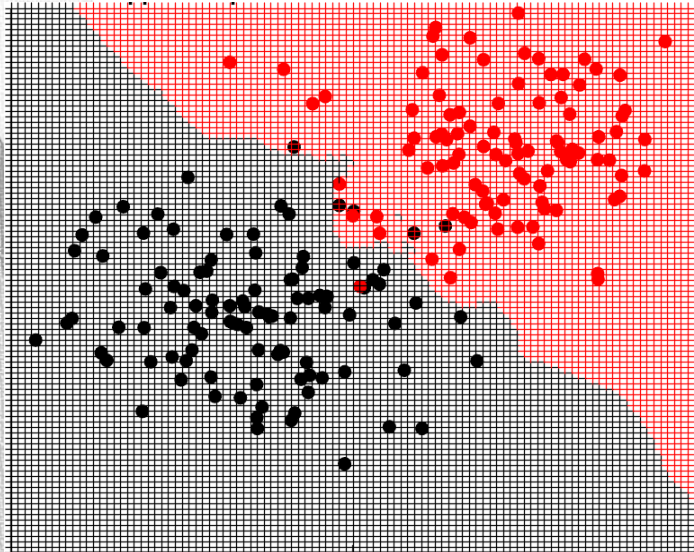
k=1



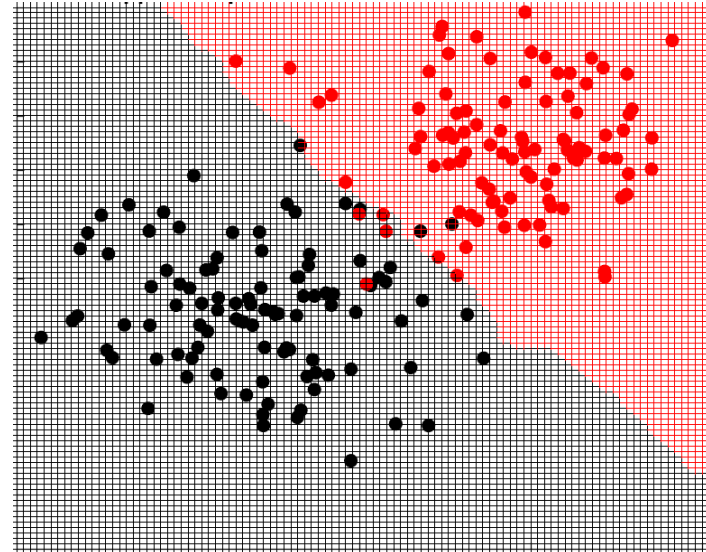
k=3



k=5



k=11



Dilemme biais/variance

k faible

- ➔ Bonne résolution des frontières entre classe
- ➔ Très sensible aux échantillons de la base de référence
- ➔ Petit biais Grande variance

k grand

- ➔ Mauvaise résolution des frontières entre classe : lissage des frontières
- ➔ Peu sensible aux échantillons de la base de référence
- ➔ Grand biais Faible variance

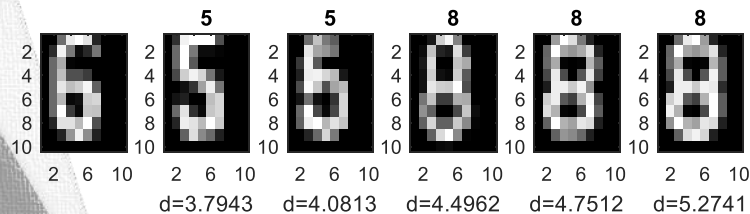
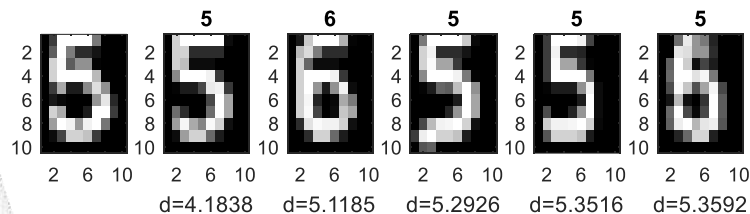
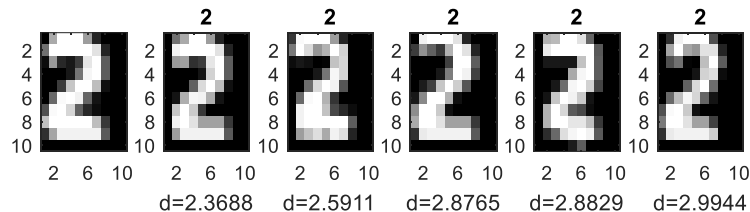
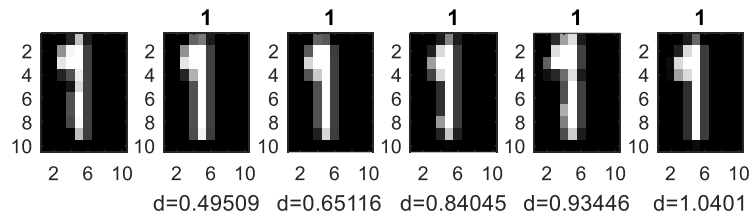
Comment choisir k ?

3 bases de données

- ➔ Base de référence où sont stockés les exemples
- ➔ Base de validation utilisée pour optimiser k
- ➔ Base de test pour évaluer les performances

Exemple en reconnaissance de caractères

- 800 exemples dans la base de références
- Chaque exemple est de dimension 100 (codage rétinien 10x10)



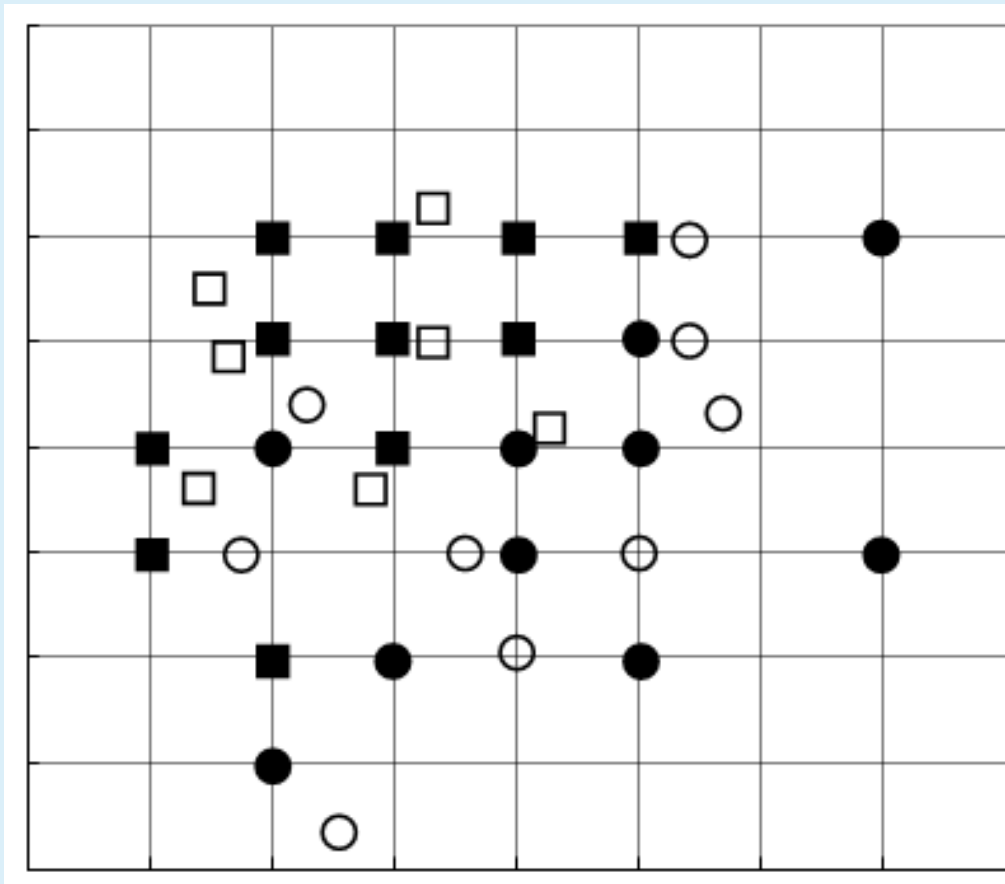
Avantages :

- Pas d'hypothèses
- Simple à mettre en œuvre
- Incrémental

Inconvénients :

- Quantité de calculs quasi-proportionnelle au nombre d'exemples
- Pas d'extraction d'information utile

Exercice

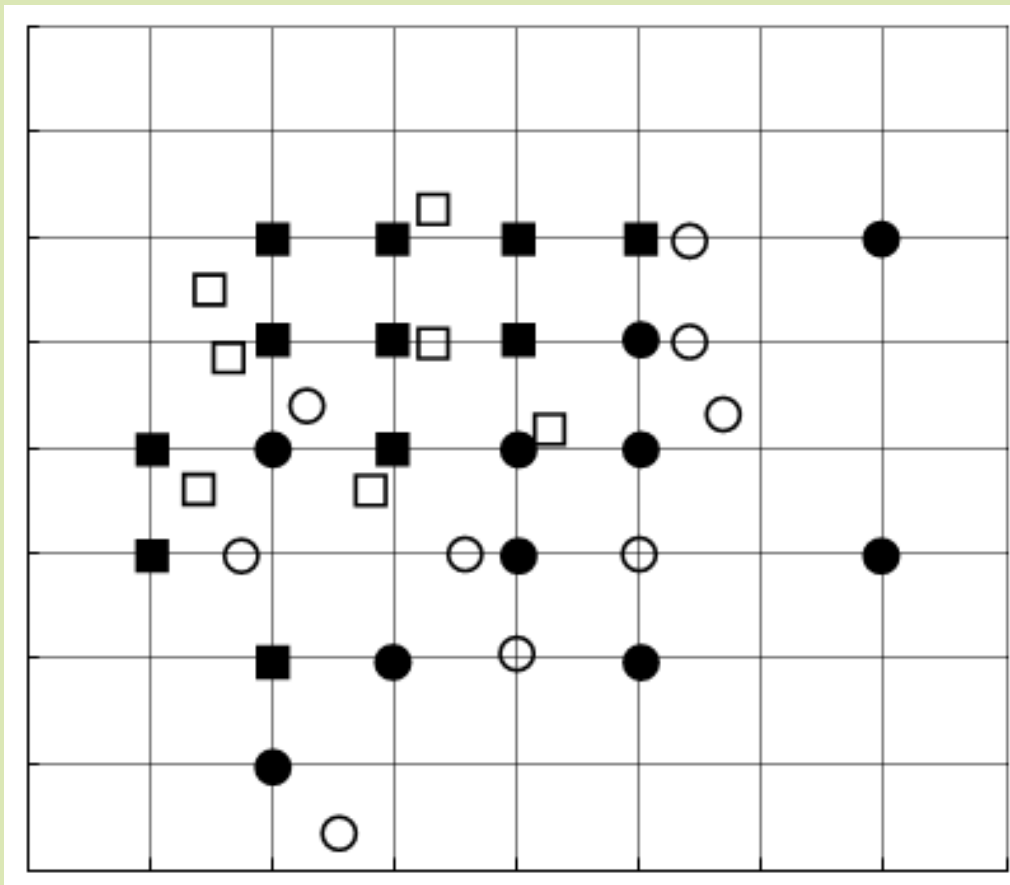


Noir : base de référence

Blanc : base de test

Donner la matrice de confusion avec l'algorithme du 1ppv

Exercice



Noir : base de référence

Blanc : base de test

	□	○
□	6	1
○	2	7

Accélération des k-PPV

2 solutions

Réduction de la dimension de chaque exemple

- ACP
- LDA

Réduction de taille de la base de référence

- On ne représente plus chaque classe que par sa moyenne
- Génération de prototypes : LVQ



Dilemme robustesse/accélération

Algorithme des KPPV

Approche 'nearest mean'

Représentation de chaque classe par sa moyenne μ_c

→ distances euclidiennes $d_E(x, \mu_c)$ entre l'exemple à classe x et les centres μ_c

$$d_E(x, c) = (x - \mu_c)^T (x - \mu_c)$$

→ L'exemple est classé à la classe dont le centre est plus proche au sens de la distance euclidienne

Représentation de chaque classe par sa moyenne μ_k et sa matrice de de covariance Σ_c

→ distances de Mahalanobis $d_M(x, c)$ entre l'exemple à classe x et les centres μ_c

$$d_M(x, c) = (x - \mu_c)^T \Sigma_c^{-1} (x - \mu_c)$$

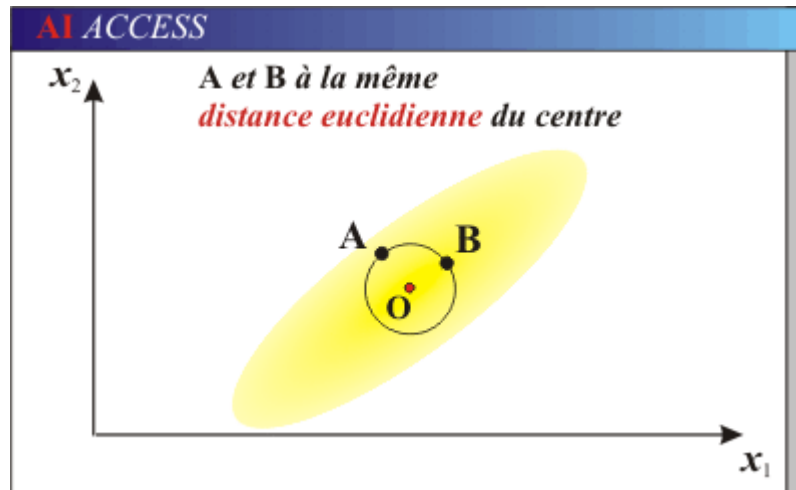
→ L'exemple est classé à la classe dont le centre est plus proche au sens de la distance de Mahalanobis

Rq : Si Σ =Identité, on retrouve la distance euclidienne

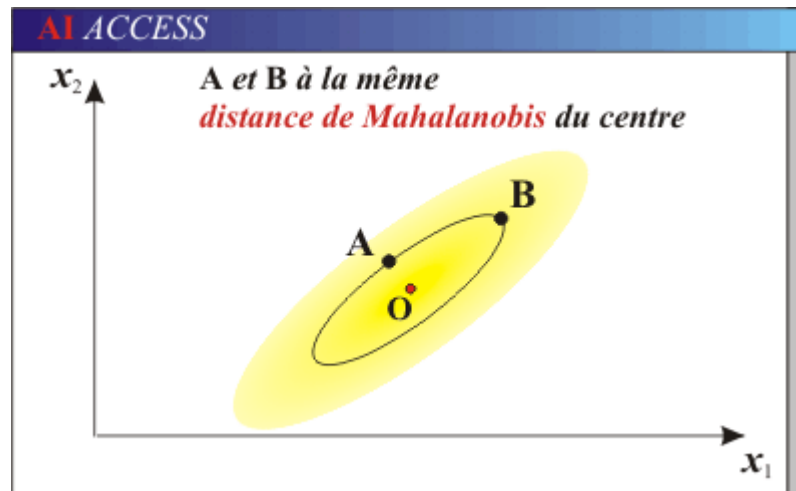
Algorithme des KPPV

Approche 'nearest mean'

Comparaison – distance euclidienne – distance de Mahalanobis



Les deux points A et B sont à la même distance euclidienne de O
➔ Pas logique

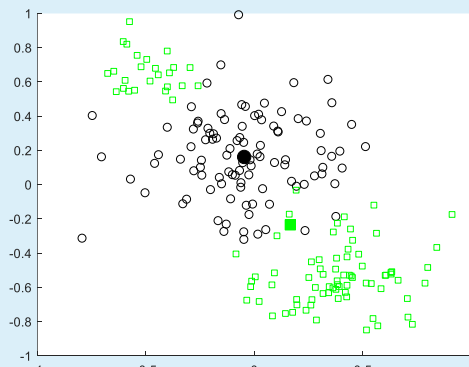
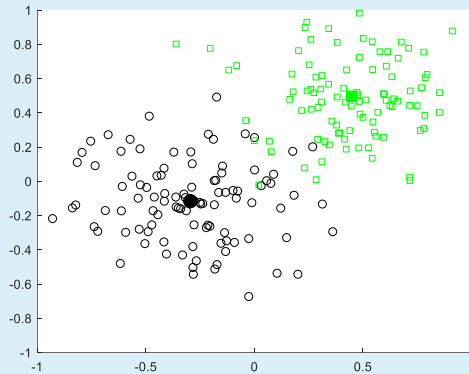
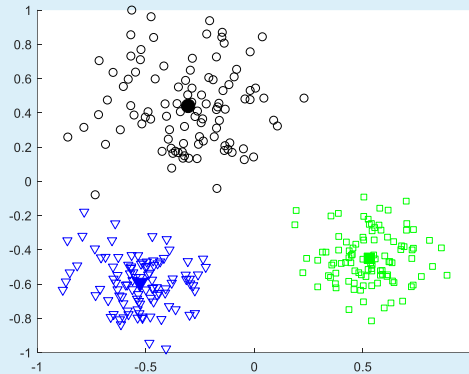


Les deux points A et B sont à la même distance de Mahalanobis de O

Algorithme des KPPV

Approche 'nearest mean'

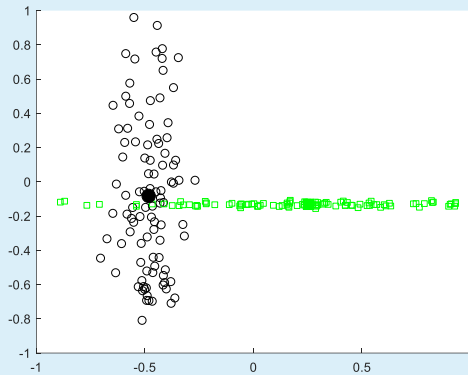
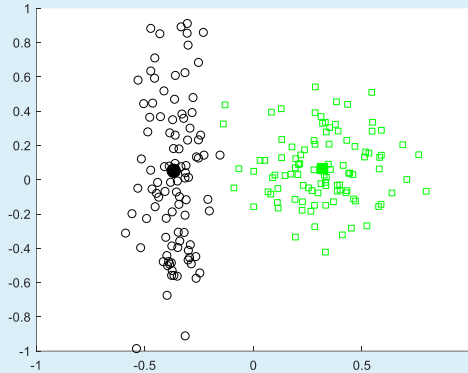
Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

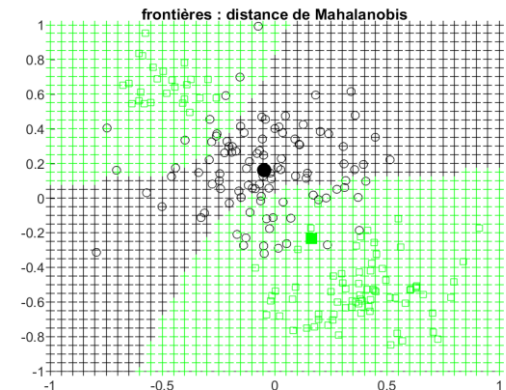
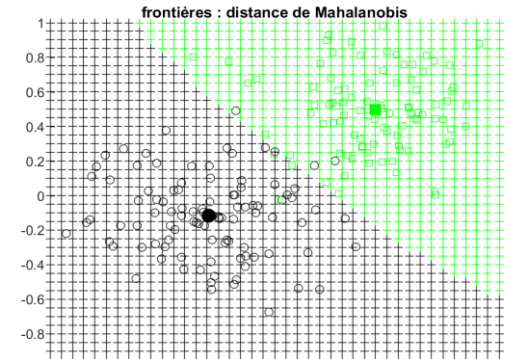
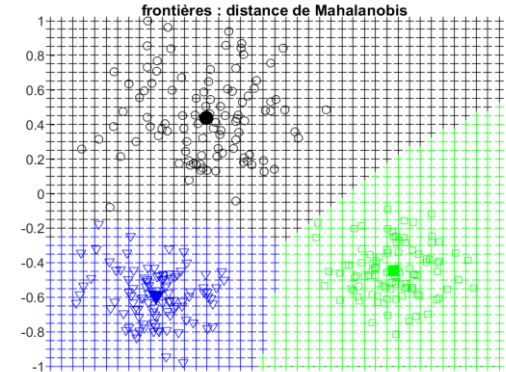
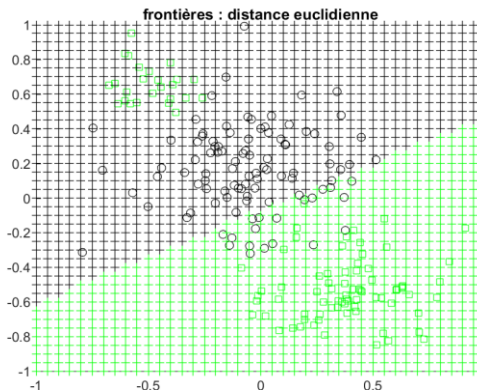
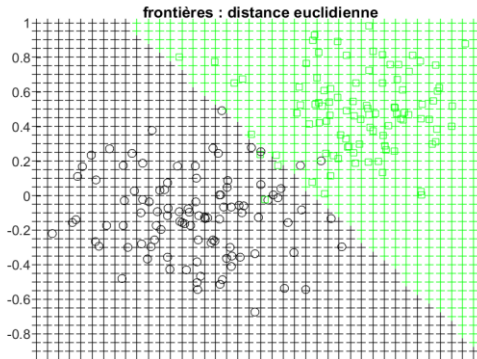
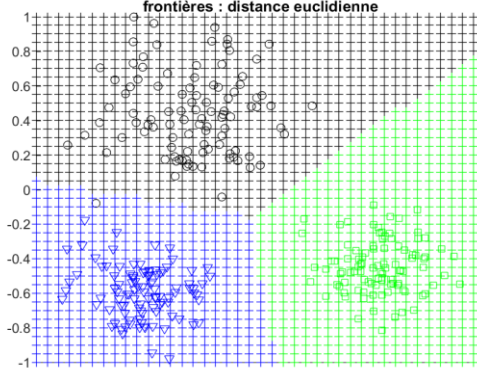
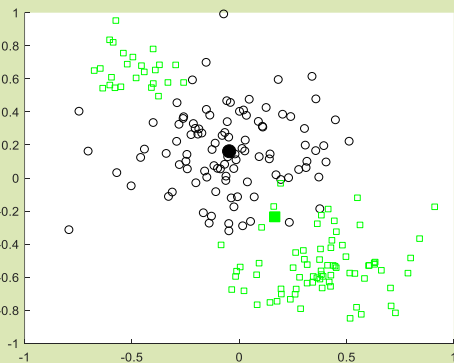
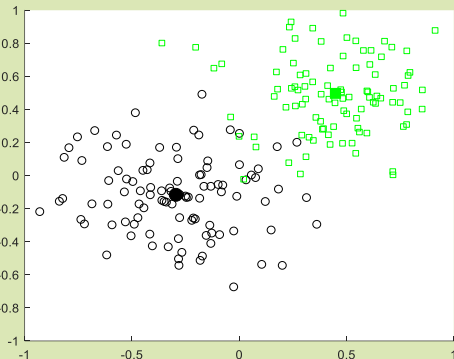
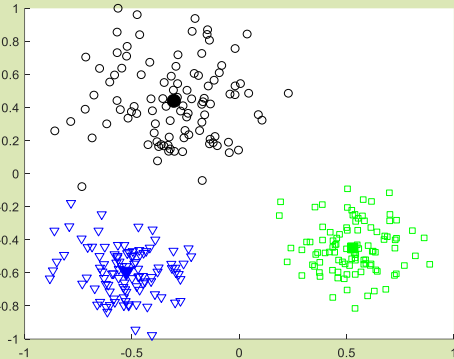
Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

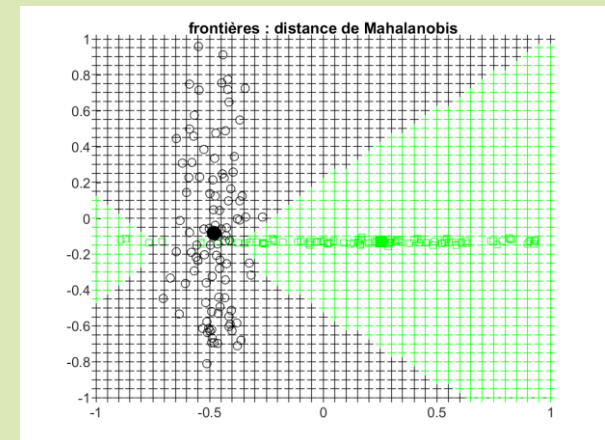
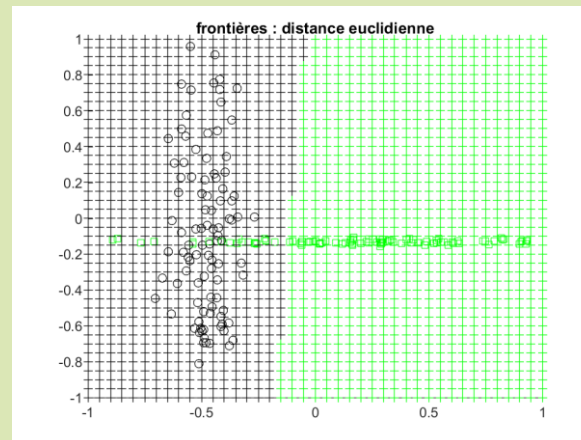
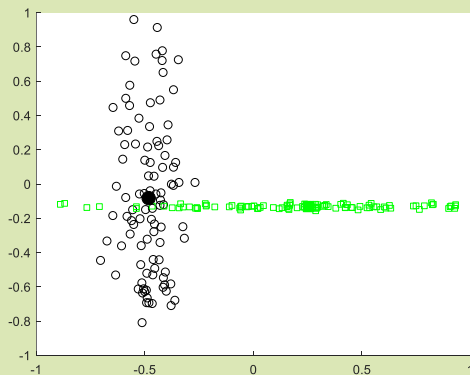
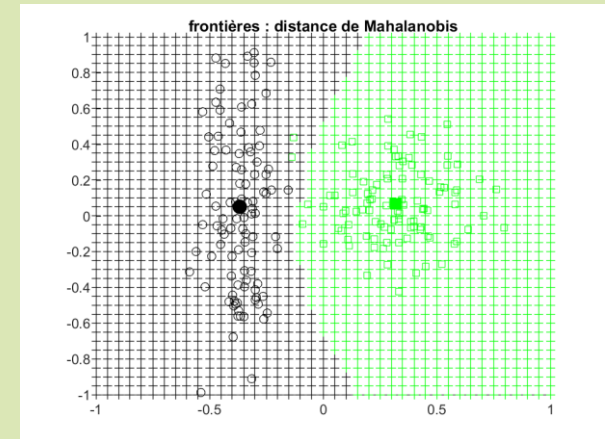
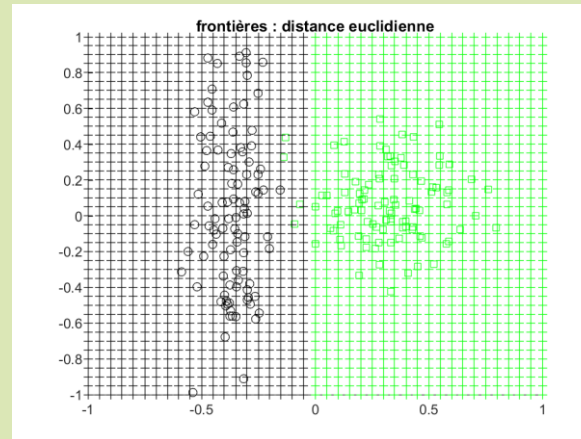
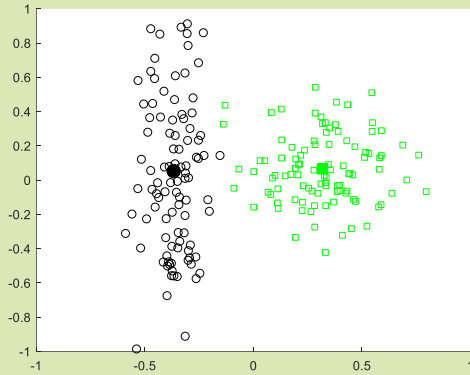
Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

Exercice : tracer les frontières entre classes avec l'approche nearest mean associée à la distance euclidienne puis la distance de Mahalanobis



Exercice:

Supposons que l'on ait un problème à 3 classes, de dimension 2. Sur la base de référence, on a estimé la moyenne et la matrice de covariance de chaque classe :

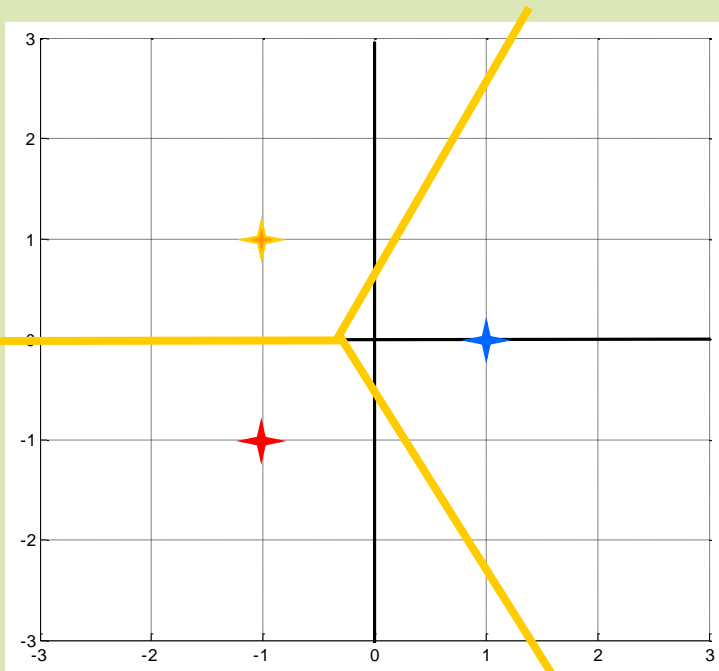
$$\begin{aligned} m1 &= \begin{pmatrix} -1 \\ 1 \end{pmatrix} & m2 &= \begin{pmatrix} -1 \\ -1 \end{pmatrix} & m3 &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ \Sigma1 &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \Sigma2 &= \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} & \Sigma3 &= \begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix} \end{aligned}$$

1. Tracer les frontières entre les classes en utilisant l'algorithme *nearest-mean* et la distance euclidienne.
2. Tracer les frontières approximatives entre les classes en utilisant l'algorithme *nearest-mean* et la distance de Mahalanobis .

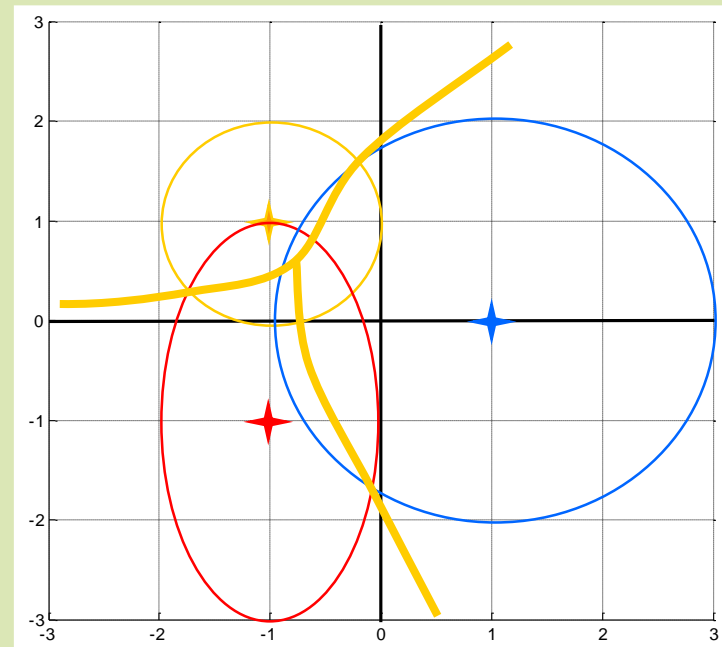
Algorithme des KPPV

Approche 'nearest mean'

Distance euclidienne



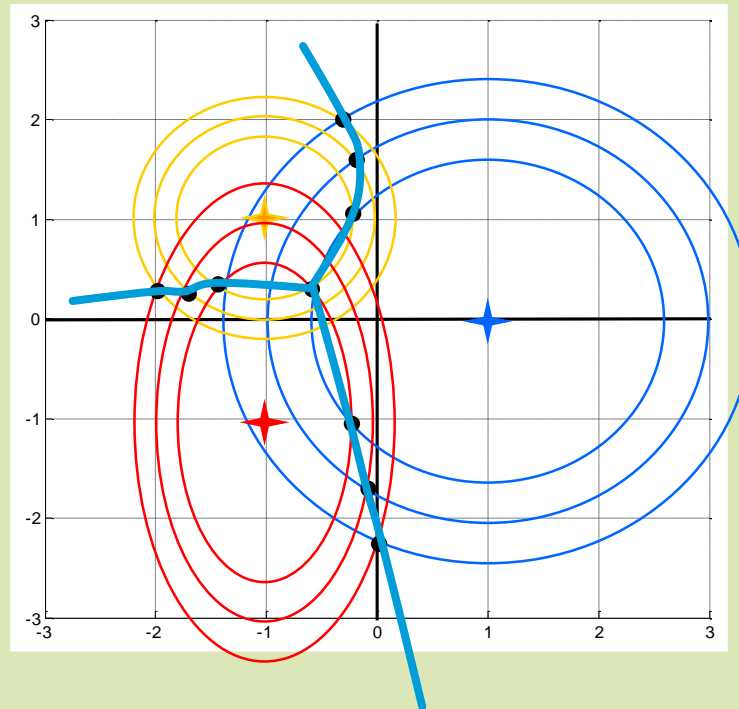
Distance de Mahalanobis



Algorithme des KPPV

Approche 'nearest mean'

Distance de Mahalanobis



Exercice:

Supposons que l'on ait un problème à 2 classes, de dimension 2. Sur la base de référence, on a estimé la moyenne et la matrice de covariance de chaque classe:

$$m1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad m2 = \begin{pmatrix} 3 \\ 0 \end{pmatrix} \quad \Sigma1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Sigma2 = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}$$

3. Considérons le point $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$. Estimer la distance de ce point à chaque classe en utilisant l'approche nearest mean et la distance euclidienne puis la distance de Mahalanobis

Exercice:

Supposons que l'on ait un problème à 2 classes, de dimension 2. Sur la base de référence, on a estimé la moyenne et la matrice de covariance de chaque classe:

$$m1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad m2 = \begin{pmatrix} 3 \\ 0 \end{pmatrix} \quad \Sigma1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Sigma2 = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}$$

3. Considérons le point $x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. Estimer la distance de ce point à chaque classe en utilisant l'approche nearest mean et le distance euclidienne puis le distance de Mahalanobis

Avec la distance euclidienne,

$$d1 = (x - m1)^T (x - m1) = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 2$$

$$\text{et } d2 = \begin{pmatrix} -2 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \end{pmatrix} = 5$$

➔ Le point appartiendrait à la classe 1

Avec la distance de Mahalanobis,

$$d1 = (x - m1)^T \Sigma1^{-1} (x - m1) = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 2$$

$$\text{et } d2 = \begin{pmatrix} -2 & 1 \end{pmatrix} \begin{pmatrix} 0.2 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \end{pmatrix} = \begin{pmatrix} -0.4 & 1 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \end{pmatrix} = 1,8$$

➔ Le point appartiendrait à la classe 2

Sélection des références LVQ

LVQ : Learning Vector Quantization

- Méthode **supervisée**, itérative :
- Génération d'un ensemble de prototypes quasi optimaux
- ➔ Minimise la variance intra-classe
- ➔ Maximise variance inter-classe

1.Initialisation aléatoire des prototypes (noyau)

2.Pour chaque exemple x , trouver le prototype p le plus proche

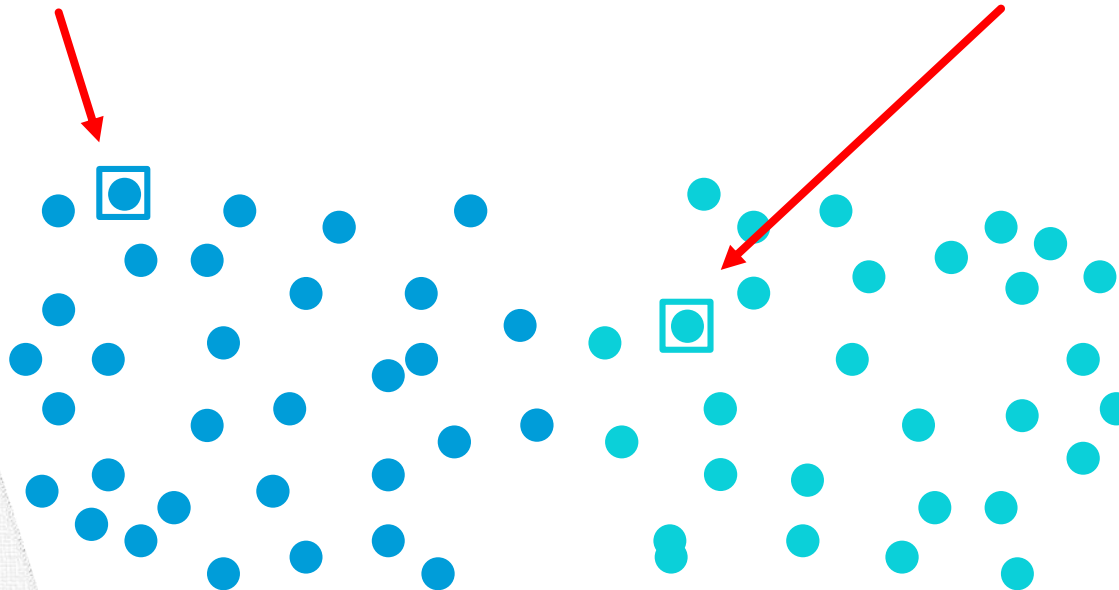
- Si p et x sont de la même classe, rapprocher p de x
- sinon, éloigner p de x

$$p(t+1) = p(t) \pm a(t)[x - p(t)]$$

où $a(t)$: pas d'apprentissage

3.Retour en 2 ou arrêt

1. Initialisation aléatoire des prototypes (noyau) de chaque classe



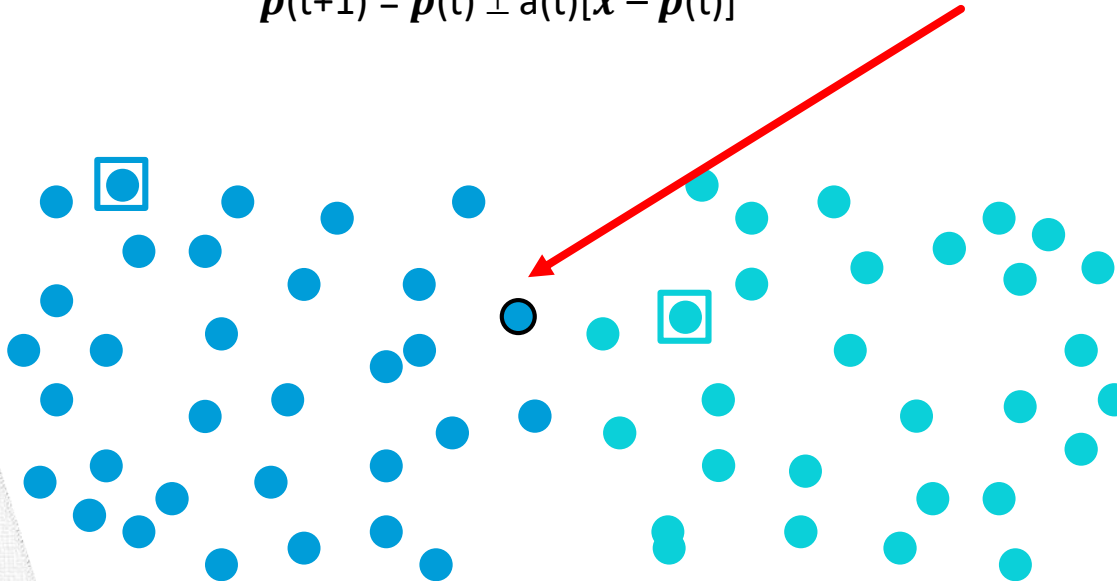
Algorithme des KPPV

LVQ

Pour chaque exemple x , trouver le prototype p le plus proche

- Si p et x sont de la même classe, rapprocher p de x
- Sinon, éloigner p de x

$$p(t+1) = p(t) \pm a(t)[x - p(t)]$$



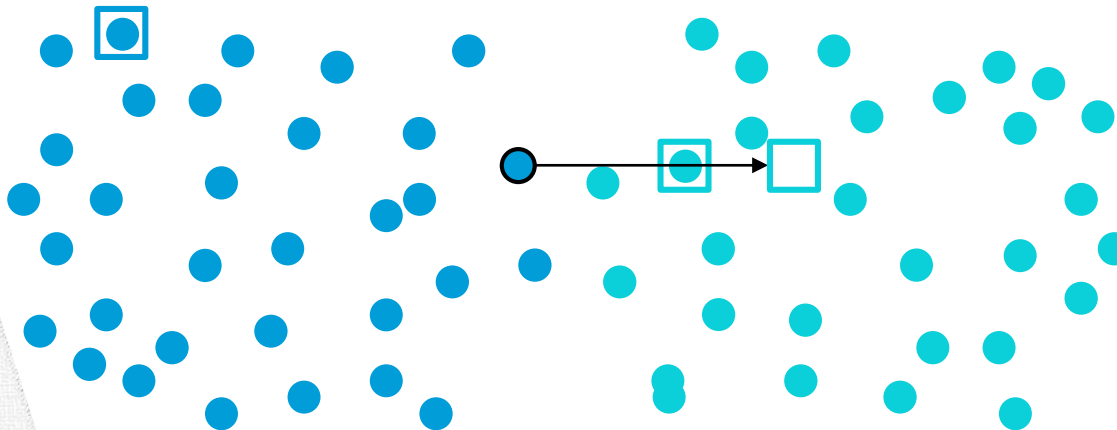
Algorithme des KPPV

LVQ

Pour chaque exemple x , trouver le prototype p le plus proche

- Si p et x sont de la même classe, rapprocher p de x
- Sinon, éloigner p de x

$$p(t+1) = p(t) \pm a(t)[x - p(t)]$$



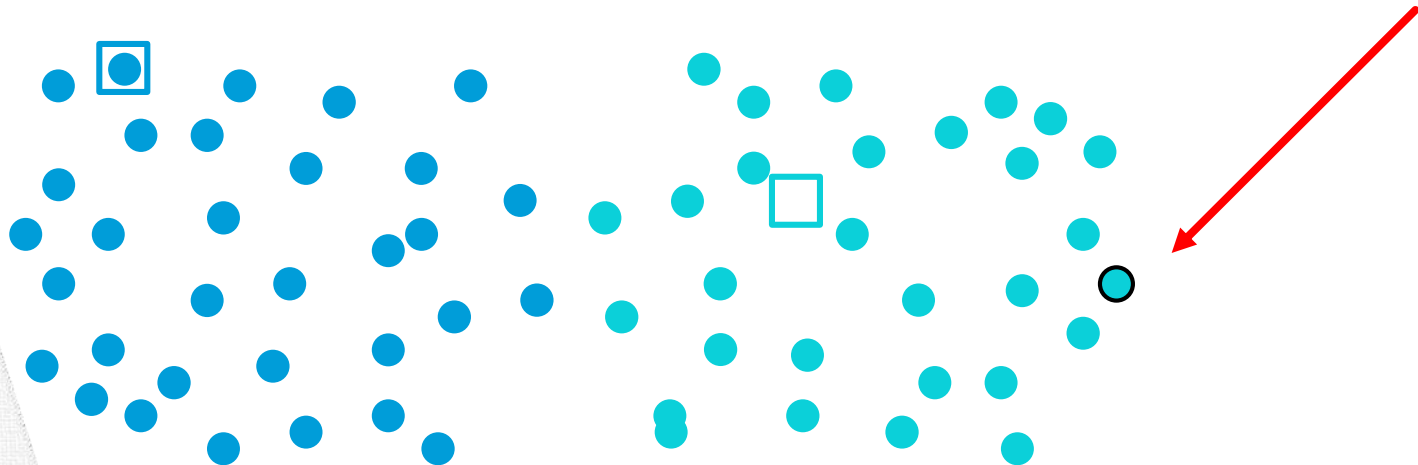
Algorithme des KPPV

LVQ

Pour chaque exemple x , trouver le prototype p le plus proche

- Si p et x sont de la même classe, rapprocher p de x
- Sinon, éloigner p de x

$$p(t+1) = p(t) \pm a(t)[x - p(t)]$$



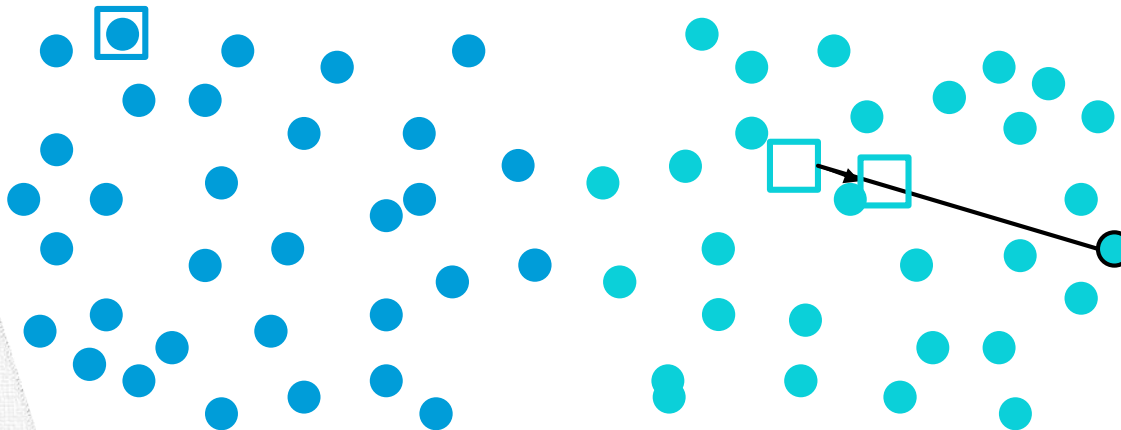
Algorithme des KPPV

LVQ

Pour chaque exemple x , trouver le prototype p le plus proche

- Si p et x sont de la même classe, rapprocher p de x
- Sinon, éloigner p de x

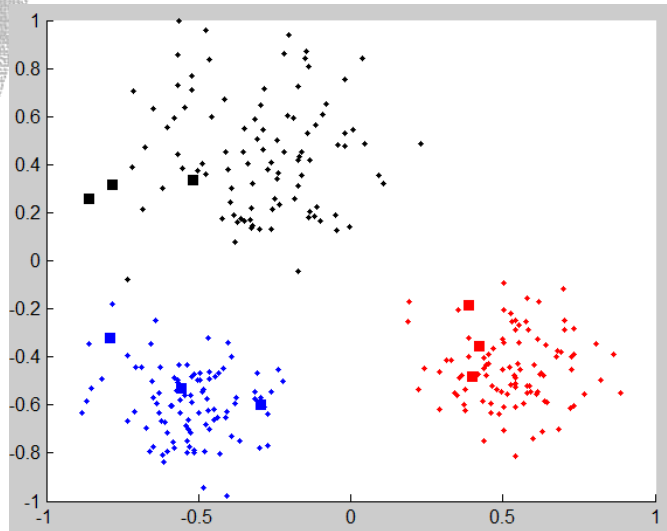
$$p(t+1) = p(t) \pm a(t)[x - p(t)]$$



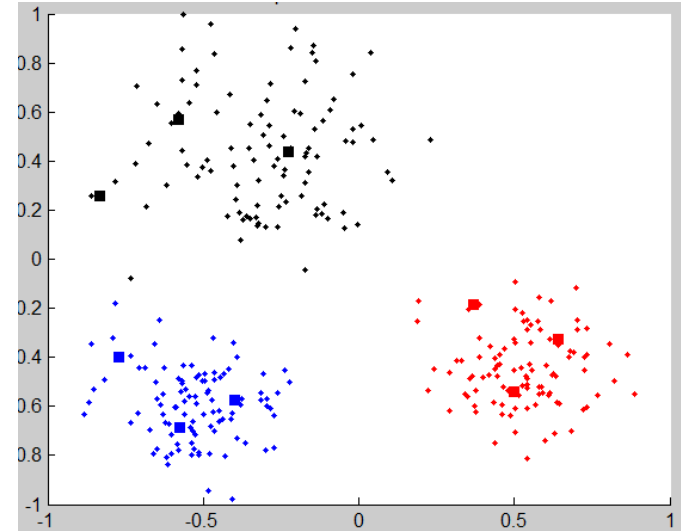
Algorithme des KPPV

LVQ

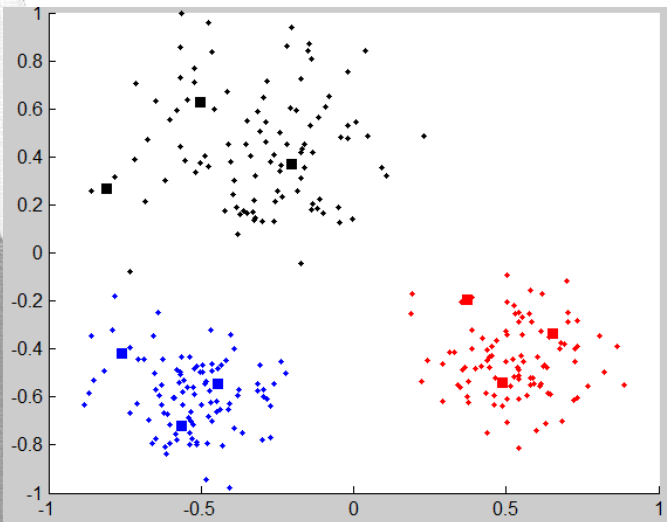
LVQ, Itération 1



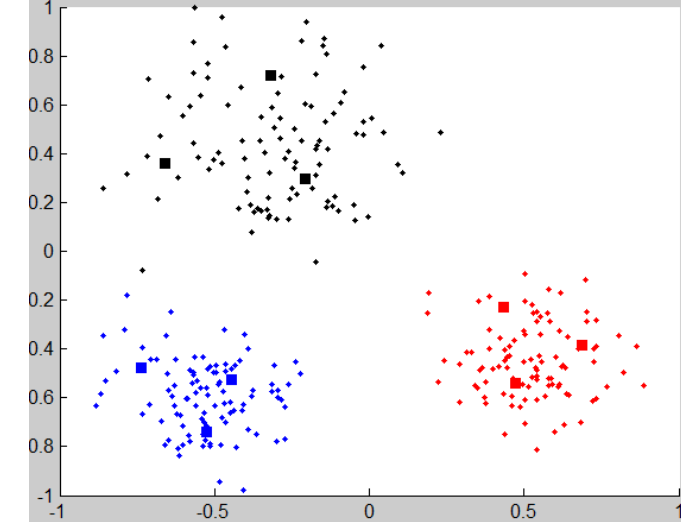
LVQ, Itération 2



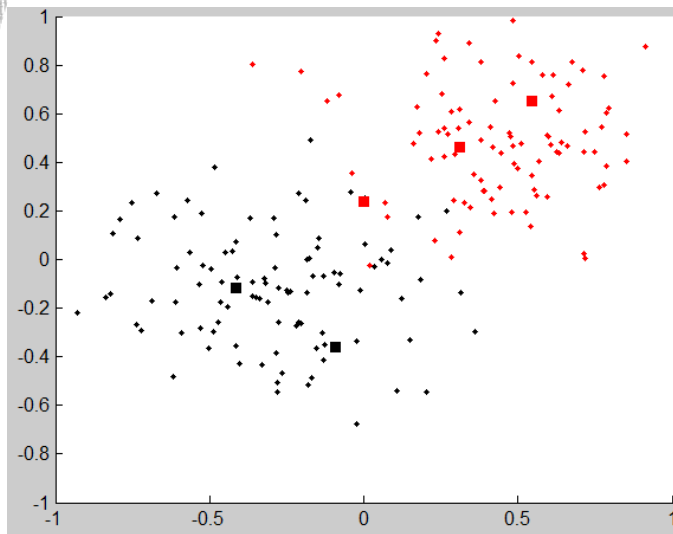
LVQ, Itération 3



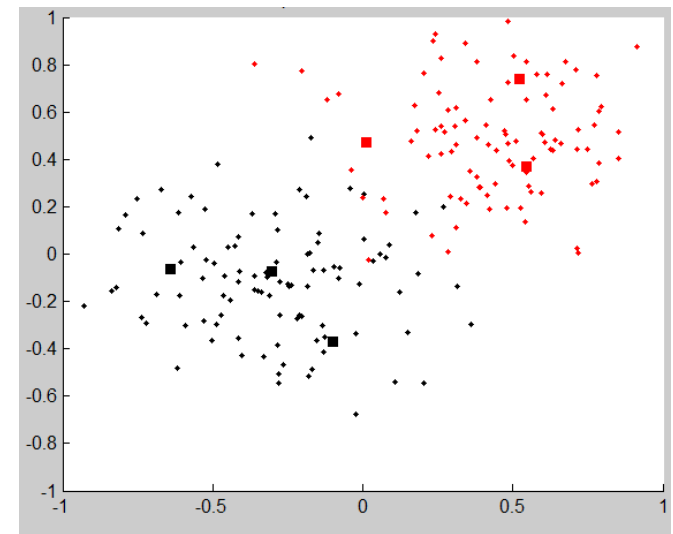
LVQ, Itération 10



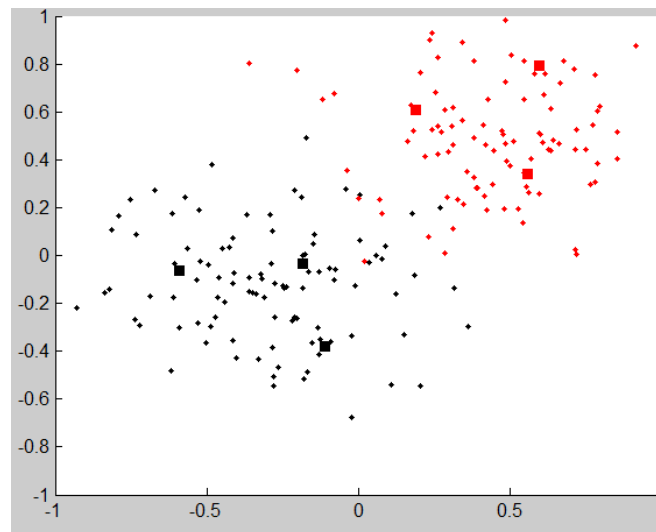
LVQ, Itération 1



LVQ, Itération 2



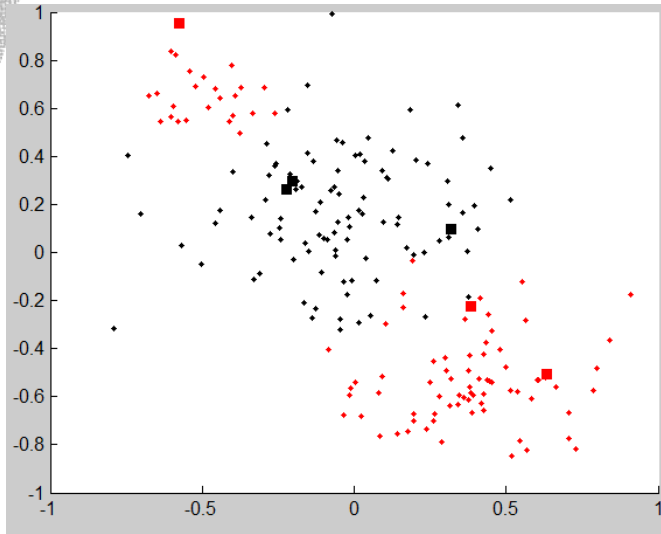
LVQ, Itération 10



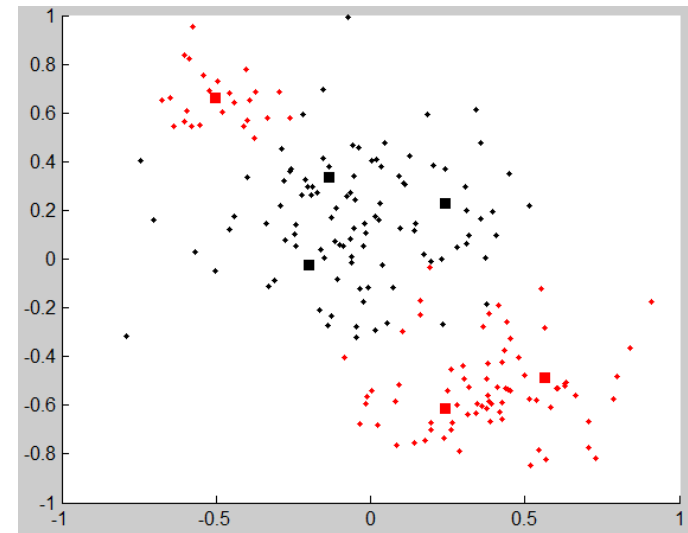
Algorithme des KPPV

LVQ

LVQ, Itération 1



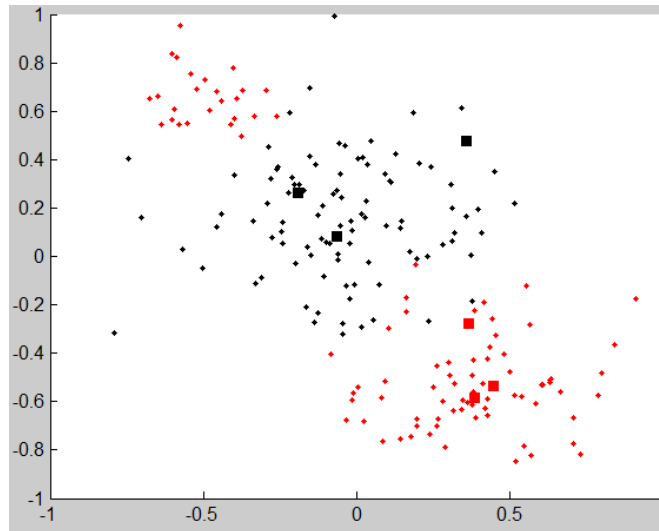
LVQ, Itération 10



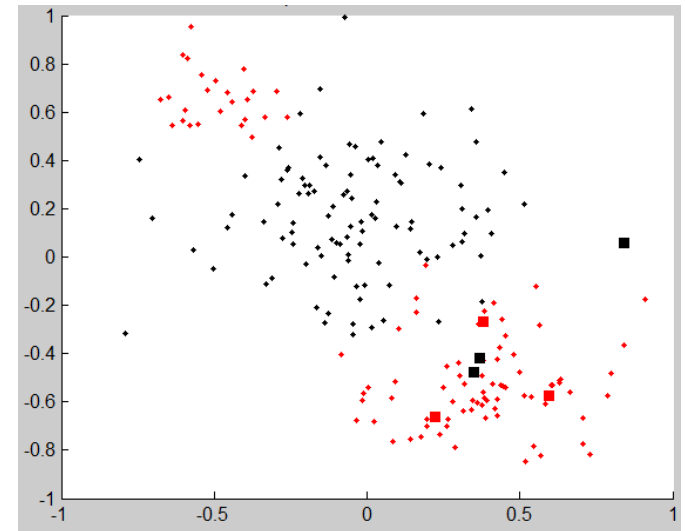
Algorithme des KPPV

LVQ

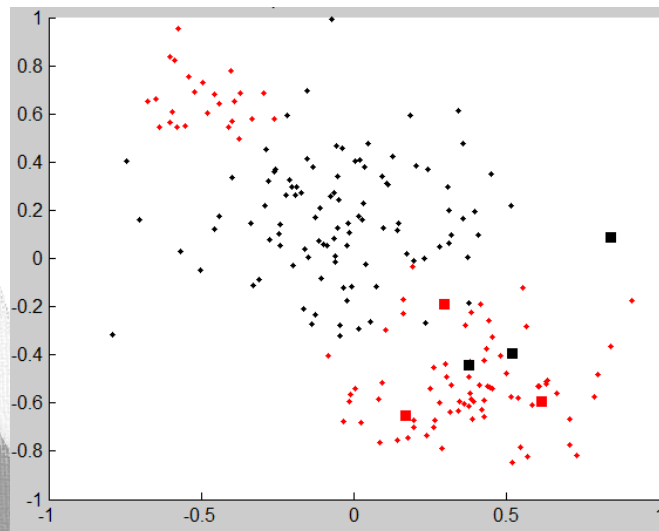
LVQ, Itération 1



LVQ, Itération 2



LVQ, Itération 3



LVQ, Itération 10

